Efficient Indexing Techniques for Biometric Databases

THESIS

Submitted in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

by

Geetika Arora

ID No. 2016PHXF0406P

Under the Supervision of

Dr. Kamlesh Tiwari

Assistant Professor, Department of Computer Science and Information Systems

Birla Institute of Technology and Science, Pilani, Pilani Campus, India

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

Pilani Campus, Rajasthan, India

June, 2022

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

PILANI CAMPUS, RAJASTHAN, INDIA

CERTIFICATE

This is to certify that the thesis entitled, **"Efficient Indexing Techniques for Biometric Databases"** and submitted by **Ms. Geetika Arora** ID No. **2016PHXF0406P** for the award of Ph.D. Degree of the institute embodies original work done by her under my supervision.

____

Signature of the Supervisor

Name : **Dr. Kamlesh Tiwari**

Designation : **Assistant Professor, Department of CSIS,**

BITS Pilani, Pilani Campus, India

Date: __/__/____

Dedicated to

My Beloved Parents

&

Brother (Gunjan)

ACKNOWLEDGEMENTS

I want to express my heartfelt gratitude to my thesis supervisor Dr. Kamlesh Tiwari, for his valuable guidance and continuous encouragement throughout my Ph.D. duration. Under his inspiring guidance, my thought process and creative thinking got matured. His knowledge and endless pursuit for excellence inspired me to work harder. His simplicity and kindness, attention to detail, hard work, and patience have set an example for me, which possibly, I hope to match someday.

I am also thankful to Dr. Aditya Nigam (Professor, IIT Mandi, India) for his guidance and valuable suggestions in my research work. I want to express my gratitude to Prof. Sudeept Mohan, Prof J.P. Mishra, Prof. Mukesh Rohil, Prof. Poonam Goyal, and Prof. Amit Dua for their support and cooperation during my research. I feel fortunate to have my Doctoral Advisory Committee (DAC) members as Prof. Navneet Goyal and Prof. Yashvardhan Sharma whose constant encouragement and support improved my research work at various stages.

I would also like to extend my sincere gratitude to my parents (Mr. Vijay Arora and Mrs. Shashi Arora) for believing in me more than myself. Their constant support and tremendous understanding kept me motivated throughout my Ph.D. and my life in general. I would also like to thank my brother Mr. Gunjan Arora for being a sympathetic ear and always encouraging me. I am deeply indebted to my entire family for their trust in me that made me confident enough to accomplish this journey successfully. I could not have completed this thesis without my friends who provided stimulating discussions and happy distractions to rest my mind outside my research. I am also thankful to the CSIS department office staff Mr. Sanwar Mal who helped me with all the formalities.

Place: BITS Pilani, Pilani Campus, India
Date: __/__/____ **(Geetika Arora)**

ABSTRACT

Identification of a query biometric sample aims to establish its identity by comparing it with all the templates stored in the database and determining the most similar one. The process is computationally expensive and becomes difficult to be carried out in a reasonable time as the size of the database increases. To get the response in real-time, the process needs to be accelerated, especially for large databases. Biometric database indexing refers to narrowing down the search space for the identification of a query sample. This is accomplished by constructing a feature vector from a biometric sample and associating it with an index in the index table. During retrieval, the most similar index to the feature vector of the query sample is selected and the candidates lying in that index are fetched for identification. This ensures that the search space is small as compared to the total database size. This thesis proposes biometric database indexing techniques for four modalities $viz.$ iris, palmprint, finger knuckleprint and fingerprint. The true identity of the query biometric sample is expected to be determined by retrieving only a small percentage of the database (low penetration rate) with high confidence (high hit rate).

For Iris databases, we have proposed a novel indexing technique, called IrisIndexNet. Iris is a ring of tissue found around the pupil. It is responsible for controlling the amount of light entering the pupil. IrisIndexNet is a custom-designed Siamese-based network that learns iris-specific features. The network utilizes larger filters to effectively capture the variance present in different iris samples of the same subject while making the learned features corresponding to different class samples distant in the embedding space. It ensures learning low-level textures effectively, which are the vital discriminating features for the iris images. The learned features are subsequently clustered using k-means and agglomerative clustering to generate an index table. The proposed technique is tested on CASIA Interval and CASIA Lamp database.

We propose PalmHashNet for indexing the palmprint databases. Palmprint is an impression acquired from the inner part of the hand lying between wrist and fingers. The acquired palmprint images are fed to the feature extraction network which is initially pre-trained using the softmax loss. The softmax loss ensures high inter-class disparity but does not ensure high intra-class compactness. Therefore, a margin is added to the softmax loss to minimize the intra-class distance between samples belonging to the same class. The network outputs a 512-dimensional distinct compact feature embedding corresponding to every palmprint sample. k-means and locality sensitive hashing (LSH) are investigated for index table creation. The proposed technique offers probabilistic guarantees for query identification in the selected bin. Experiments are conducted on four widely used palmprint databases $viz.$ CASIA, IITD-Touchless, Tongji-Contactless and Hong Kong Polytechnic University Palmprint II (PolyU II) database. To accelerate the identification process for finger knuckleprint (FKP) samples, we present FKPIndexNet, that learns similarity-preserving hash codes for generating index table. FKP refers to the impression obtained from the outer surface around the phalangeal joint of a finger. In our knowledge, this is the first work to extract deep features from FKP images that are further used for index table generation. Firstly, the dorsal finger images are given to the proposed *FKPSegNet* which segments out the region of interest (RoI). Next, the feature extraction network is trained using the RoI images of the FKP database. A custom loss function has been proposed ensuring that the learned latent representations have high intra-class and low inter-class similarity. It ascertains that the index space distribution is regularized to be similar to the uniform distribution. The proposed feature extraction network outputs a $512-$dimensional feature embedding corresponding to every FKP sample. The learned feature vectors are associated with an index and an index table is generated. Three techniques $viz.$ k-means clustering, BallTree hashing and locality sensitive hashing with nearest neighbor search have been explored for index table generation. The proposed technique is tested on two publicly available benchmark FKP databases $viz.$, PolyU-FKP and IITD-FKP.

An indexing technique for fingerprint databases is proposed by utilizing a feature vector constructed using the directional and spatial relationship between core and minutiae points. A fingerprint is a skin pattern acquired from the tip of a finger. It is a widely accepted biometric trait for access control. The proposed technique constructs a feature vector for each fingerprint sample by encoding a spatial and directional relationship between minutiae and core point. Therefore, the first step in the proposed technique is to extract the locations of the core and minutiae points. A novel deep learning based architecture is proposed for locating the coordinates of the core point in a given fingerprint image. Minutiae points are extracted using a Mindtct library given by NIST Biometric Image Software (NBIS). The proposed feature vector, Coaxial Gaussian Track Code (CGTC), is constructed for each minutiae point in the fingerprint and is inserted in the index table exactly once, making the time complexity linear. The proposed approach has been tested on FVC2002_DB2a and FVC2004_DB1a databases.

A biometric database indexing technique is considered to be effective if it is able to achieve a higher hit rate at a lower value of penetration rate. *IrisIndexNet* requires 2.254% and 0.008% penetration rate at 99% hit rate on CASIA Interval and CASIA Lamp database respectively. In other words, one needs to search only 2.254% and 0.008% of the considered datasets to be 99% sure about the presence of a sample in the database. *PalmHashNet* achieved a penetration rate of 0.022%, 1.032%, 4.555% and 0.39% at 100% hit rate on CASIA, IITD-Touchless, Tongji-Contactless and Hong Kong Polytechnic University Palmprint database. Therefore, to find the true match of a query sample with 100% confidence, it is required to look for less than 1% of the CASIA and PolyU II database and 1.03% and 4.55% of the IITD-Touchless and Tongji Contactless database respectively. The proposed indexing technique for finger knuckleprint database (*FKPIndexNet)* achieved 100% of hit rate at the penetration rate of only 3.42% and 0.32% on PolyU-FKP and IITD FKP database, respectively. Lastly, the fingerprint indexing technique requires a penetration rate of 0.86% for a 100% hit rate for identification on the FVC2004_DB1a database. The obtained results for the proposed indexing techniques require only a small percentage of the database instead of the whole for identification of a query biometric sample. The proposed techniques output a fixed size candidate list for comparison with the query biometric sample thereby, making identification a constant time operation.

Contents

List of Abbreviations

AFIS :	Automatic Fingerprint Identification System
AUC :	Area Under the Curve
BWT :	Burrows Wheeler Transform
CGTC :	Coaxial Gaussian Track Code
CMC :	Cumulative Matching Curve
CNN :	Convolutional Neural Network
COSDISH :	Column Sampling Based Discrete Supervised Hashing
CP :	Contracting path
DI :	Decidability Index
DITOM :	Densely Infinite-to-one Mapping
DL :	Deep Learning
EER :	Equal Error Rate
EP :	Expansive path
FA :	False Acceptance
FAR :	False Acceptance Rate
FKP :	Finger-knuckle-print
FKPIndexNet :	Finger knuckleprint Indexing Network
FR :	False Rejection
FRR :	False Rejection Rate
FVC :	Finger Verification Competition
GAP :	Global Average Pooling
GORP :	Gradient Ordinal Relation Pattern
HR :	Hit Rate
ID :	Identifier
IDN :	Iris Descriptor Network

IR :	Identification rate
IrisIndexNet :	Iris Indexing Network
JS-divergence :	Jensen–Shannon Divergence
KL-Divergence :	Kullback–Leibler Divergence
LBP :	Local Binary Pattern
LLDP :	Local binary pattern-based feature descriptor
LoD :	Low Order Delaunay
LoG :	Laplacian of Gaussian
LSH :	Locality sensitive hashing
MCC :	Minutia Cylinder Code
MLN :	Macro-Localization Network
MRN :	Micro-Regression Network
OP :	Orientation pattern
ORB :	Oriented FAST and Rotated Brief
PalmHashNet :	Palmprint Hashing Network
PCM :	Polar Complex Moment
PolyU II :	Hong Kong Polytechnic University Palmprint II Database
POP :	Principal orientation patterns
PR :	Penetration rate
QLT :	Quantized Lookup Table
RLOC :	Robust line orientation code
ROC :	Receiver Operating Characteristic
RoI :	Region of Interest
SIFT :	Scale Invariant Feature Transform
SURF :	Speeded Up Robust Features
SVM :	Support Vector Machine
TDR :	True Detection Rate

List of Tables

List of Figures

Chapter 1

Introduction

Personal authentication plays an important role in controlling access to high-security areas and sensitive documents. Traditional authentication methods are either *possession* or *knowledge* based. Possession-based authentication involves the use of a key or token owned by the user. In contrast, knowledge-based authentication requires a user's knowledge such as a PIN or password [1]. Although these methods have been accepted and widely used by society for a long time, some inherent issues are associated with them. For instance, a key or token can be misplaced by the user. If found or stolen by an imposter, the same can be used to get illegitimate access to a system. The user often creates a memorable text-based password, but it can be easily guessed by the imposter. However, a password that satisfies stringent complexity requirements of a system is forgettable. These limitations can be overcome by using personality characteristics. These characteristics are broadly termed as *biometrics* and utilize physiological or behavioral traits of an individual [2]. A biometric trait cannot be forgotten by the user and is nearly impossible to be stolen.

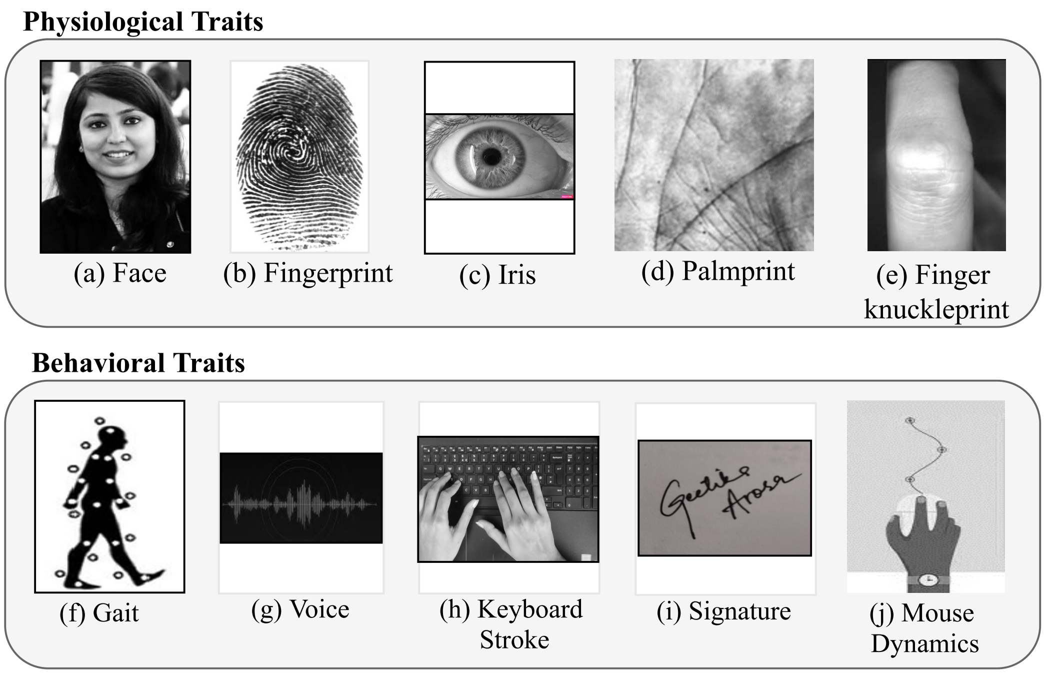


Figure 1.1: Examples of some physiological and behavioral biometric traits

1.1 Biometrics

Biometrics provides an automated way to authenticate an individual's identity by using their physiological and/or behavioral characteristics. Physiological traits refer to the characteristics that an individual physically possesses and can be acquired from the human body. Fingerprint [3, 4], palmprint [5], iris [6], face [7],finger-knuckle-print [8], hand geometry [9], hand vein pattern [10] etc. are some of the examples of physiological biometric traits. Behavioral traits refer to how an individual performs a particular action such as while signing, speaking, walking, typing etc. Therefore, signature [11], voice [12], gait [13], mouse dynamics [14] and keyboard strokes [15, 16] respectively are considered as behavioral biometric traits [1]. Examples of some physiological and behavioral biometric traits such as face, iris, palmprint, finger-knuckleprint, fingerprint, gait, voice, signature, keyboard stroke and mouse dynamics are shown in Figure 1.1.

Nowadays, biometric authentication is being used in many applications and by many organizations such as law enforcement (to identify the suspects in a crime scene), national security (to grant/revoke entry to an individual), healthcare (to identify a patient), banking and finance (to authenticate banking transactions) and commercial application (recording attendance of employees). Biometric authentication has many advantages over traditional methods of authentication. A few of these are listed below.

1. **Increased security:** Adaptation of biometrics for authenticating an individual's identity has provided a high level of security as the passwords and PINs are likely to get compromised. It is comparatively easy for a fraudster to guess someone's password than to circumnavigate a biometric authentication system.

2. **User Convenience:** The internal mechanism of a biometric authentication system may be complex but is convenient and quick from the user's point of view. For instance, using a fingerprint for unlocking the phone is faster than typing long and complex passwords. The users also do not need to remember their biometrics, unlike passwords and PINs, which are easily forgettable.

3. **Non-transferable:** Individual's biometric cannot be shared as it is intrinsic to the user. It is compulsive for the user to be physically present to give their sample at the time of authentication. This also brings an aspect of non-repudiation as the user cannot later deny his participation in the authentication.

4. **Hard to spoof:** Spoofing happens when a fraudulent person disguises himself as someone else to get access through an authentication system. Biometric traits are hard to spoof as they are unique, even among identical twins.

The utility of a biometric trait for authentication is measured based on six properties $viz.$ uniqueness, universality, circumvention, collectability, permanence, and acceptability. Uniqueness refers that the biometric traits of different persons should not be same. Universality ensures that the considered biometric trait is obtainable from the majority of the population. For example, a scar or mole may be unique to a person, but it is not found in every person. Collectability refers to the amount of effort in terms of user co-operation and extra hardware and software requirements to acquire a biometric trait. For instance, the fingerprint is relatively easy to acquire than footprint or iris. The users need to remove their shoes and socks to register their footprint sample, and iris acquisition may require extra user cooperation and hardware. Permanence measures the effect of aging on the biometric trait. A trait that remains temporally invariant or depicts minimal change is preferable. The face of a person undergoes variation with age. Acceptability refers to the level of acceptance that the population has for a given biometric trait. For example, people are comfortable providing their fingerprint samples, while a few may be hesitant to give out their facial images. Circumvention determines the degree to which a biometric trait is resilient to attack. Fingerprint, for instance, can be spoofed by showing a gummy finger to the authentication device. Gummy fingers are developed using latent fingerprints (acquired from a surface) and later put into a gelatin artificial finger mold. However, no biometric trait can fulfill all the properties mentioned above with absolute confidence. Therefore, a biometric trait that satisfies most properties with high confidence can be utilized for authentication. The selection of suitable biometric traits can also be made based on the targeted application type. The biometrics properties are described in detail as follows. Some of the most widely used physiological and behavioral biometric traits are discussed in detail as follows:

1. **Face:** Face recognition [7] refers to identifying a person using his/her face. It is one of the most common and intrusive biometric traits used for authentication. Face recognition involves three stages; 1) detecting the face from an acquired image, 2) facial feature extraction and 3) comparing the facial attributes of the acquired sample with the template(s) stored in the database. A sample of acquired face sample is shown in Figure 1.1(a).

2. **Fingerprint:** Fingerprint [2, 17] refers to an impression that can be obtained when the inner part of a finger gets touched on a surface. A sample fingerprint is shown in Figure 1.1 (b). It offers several advantages over other biometric traits such as, uniqueness, permanence and acceptability. The curves present on fingerprint, called ridge lines form unique patterns for every fingerprint. It is easy to acquire a finger-

print and extract the ridge pattern. The probability of two fingerprints to be identical is as low as 1.5×10^{-15} [18]. It has been observed that fingerprints do not undergo temporal changes. Fingerprints are the most expressible biometric traits as there exists a huge amount of discerning patterns such as loop, whorl and arch on it. Loop and whorl are called fingerprint singularities that are generally treated as global feature. Further, capturing and identifying a fingerprint is relatively inexpensive.

3. **Iris:** Iris [19] is an internally protected organ of the eye. It is located between lens and cornea as shown in Figure 1.1 (c). It is one the most suitable biometric traits for human authentication [2] as it contains rich textural information in the form of fibrous muscles, color, minutia, spots, filaments, rifts, furrows, crypts etc. Iris is found to be temporally invariant and is internally protected thus, cannot be easily duplicated [20]. It has low false-matching rate as compared to other available biometric traits [21].

4. **Palmprint:** Palmprint [5] is an impression that is acquired from the inner part of the hand lying between wrist and fingers. It consists of complex and unique patterns that are utilized as features for human authentication. The features can be classified as high resolution and low resolution based on the quality of acquired image. The low resolution features include wrinkles, texture and principal lines which can be further sub-classified as heart, head and life line. These are visible with naked eye as well. On the other hand, ridges, singular point(s) and minutiae points can only be extracted from high resolution images [22]. A sample palmprint image is shown in Figure 1.1 (d).

5. **Finger Knuckleprint:** Finger-knuckleprint (FKP) [23] refers to the impression obtained from the outer surface around the phalangeal joint of a finger, as shown in Figure 1.1 (e). The surface of FKP is rich in texture and consists of lines and creases that form a unique pattern among the population. FKP pattern is small in size and thus, it requires shorter processing time. The advantage of FKP is that the

skin pattern around an FKP is less prone to damage as it lies on the outer surface of the hand.

6. **Gait:** It is a behavioral biometric trait that refers to how an individual walks [13, 24]. Gait possesses an advantage over other physiological traits in terms that it does not require user cooperation during acquisition. It is therefore, ideal in scenarios where direct interaction or cooperation with the users is not possible such as surveillance applications. Optic flow can be used by the systems to determine gait of a person as shown in Figure 1.1 (f). But, it requires expensive hardware. Also, the gait of a person is not very distinct among the population and it may easily affected by physical and circumstantial conditions of the user such as clothing, surface, speed, type of shoes etc.

7. **Voice:** Voice [12, 25] of a person can be used to recognize the speaker as each individual possesses distinct voice characteristics such as pronunciation style, voice texture etc. Various voice properties such as inflection, cadence, frequency, nasal tone etc. are used for voice recognition. Voice recognition is socially acceptable and is non-intrusive. It does not require costly hardware as general purpose voice recorders can be used to acquire voice data of an individual. However, voice of a person gets affected due to illness, ageing, physical or emotional stress conditions etc. Another disadvantage is that voice of a person can be easily acquired from various sources such as podcasts, phone recordings etc. and can be used by an imposter to gain illegitimate access to a system.

8. **Keyboard Stroke:** Keyboard stroke [15, 16] refers to the way a person types characters on a keyboard, mobile phones or palmtops. The keyboard strokes of two individuals are unique enough to distinguish between them. The features can be extracted in the form of typing speed, elapse time between pressing two keys, the sequence of keys followed by a person while typing an uppercase character of a special character *etc.* Acquisition of keyboard stroke data is easy as it does not

need any additional hardware and administration. The user can be himself and type the characters during data acquisition.

9. **Signature:** Signatures [11] are widely used for authentication in banking organizations and law firms. Various features can be uniquely identified from the signature of a person such as angle of writing, amount of pressure applied on the paper, number of connected components, letter formation etc. A sample signature is shown in Figure 1.1 (i) However, signature of a person may change with time. It can also be easily reproduced by the imposter.

10. **Mouse Dynamics:** This refers to assessing and measuring an individual's mouse-behavior characteristics for authentication. Mouse dynamics [26] is less intrusive and does not require any additional hardware for acquisition. A user can be asked to perform some mouse operations on a computer to record their mouse dynamics for logging-in to a system. The behavior of the user is recorded in terms of mouse movements and clicks.

1.2 Modes of Operation

A biometric system works in two modes $viz.$ *enrollment* and *authentication*. Enrollment refers to the process of capturing and storing biometric information from an individual. The acquired biometric samples from the individuals are registered in a biometric database with a unique identifier. The authentication process compares a newly acquired sample, known as query or probe, with an already enrolled biometric sample(s) stored in the database. Authentication is further divided into two categories- *verification* and *identification* on the basis of the number of comparisons required. The comparison between two biometric samples is accomplished by computing a distance metric between their respective feature vectors. The system can output a similarity or dissimilarity score based on the distance metric used $i.e.$ the cosine similarity gives similarity score when two feature vectors are compared. In contrast, the euclidean distance outputs the dissimilarity score. In this work, we will refer to similarity scores whenever we are comparing two feature vectors. The similarity score is expected to be higher when two samples of the same identity are compared with each other instead of two samples belonging to different identities. A comparison is called **genuine** if two biometric samples with the same identity are compared with each other. On the other hand, if two samples belonging to different identities are compared with each other, then it is referred to as an **imposter** comparison. The feature vectors of two different samples of the same individual can be different to some extent because of the presence of illumination, occlusion, pose variation, background noise etc. The variation observed between two samples of the same individual is known as intra-class dissimilarity. While the difference found between two samples of different individuals is known as inter-class dissimilarity.

1.2.1 Enrollment

Enrollment refers to the process of registering the biometric sample of an individual with a biometric system. During enrollment, a new biometric sample is collected using an acquisition device or a sensor. The raw biometric sample may contain unwanted background noise. Therefore, the region of interest (RoI) is segmented out from the acquired image. It is followed by quality assessment of the acquired sample. Quality assessment is an indicator of the usefulness of the biometric sample for recognition. Some of the popular quality assessment techniques make use of light convolutional neural network (CNN) with the Max-Feature-Map units [27], 2D wavelets [28], face patches [29], likelihood ratio based fusion method to fuse six quality parameters [30] etc. have been proposed in literature. If the quality of the RoI is not as good as a pre-determined threshold, then the sample is re-acquired. Otherwise, the RoI is sent for pre-processing. Pre-processing aims at enhancing the RoI of the image to make authentication easier and efficient. Pre-processing is followed by feature extraction, which works towards extracting salient features from the biometric sample. The feature vector, also called a template, is stored in the biometric database with a unique identifier (ID) of the user for later use. The block diagram of an enrollment process is presented in Figure 1.2.

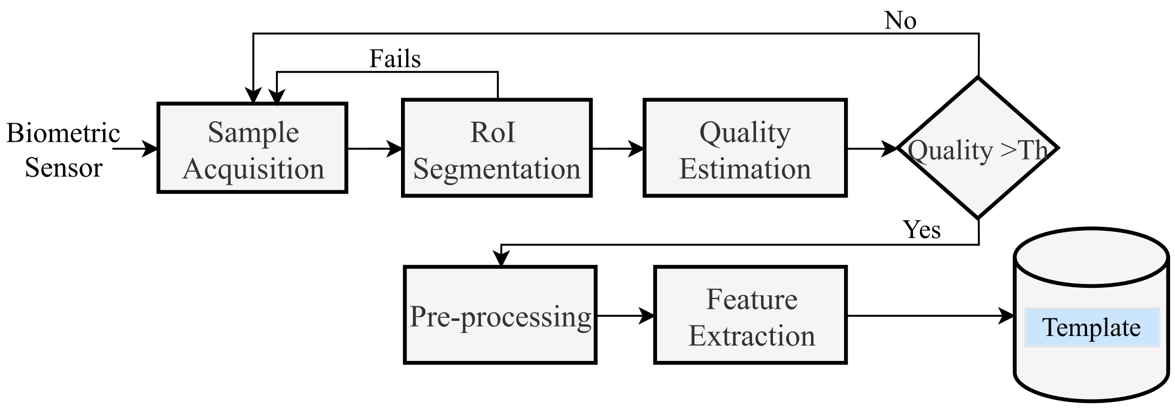


Figure 1.2: Flow chart of enrollment process.

1.2.2 Verification

The verification system validates if the identity claimed by a user is genuine or not. The input to a verification system is a newly acquired biometric sample and an identity that the user claims for the sample. The same procedure involving RoI segmentation, quality estimation and pre-processing is done on the input sample. This is followed by feature extraction of the query biometric sample. Since a claimed identity is given as an input to the verification system, the feature vector is only compared with the template stored in the database against the claimed identity. Consequently, verification requires only one-to-one comparison. In other words, if f_q denotes the feature vector of the query biometric sample and the template corresponding to the claimed identity is represented by f_{ID} then, comparison between f_q and f_{ID} is carried out by computing the similarity score using the *comparison* module. The decision module compares the obtained score with a pre-defined threshold. The comparison is treated as genuine if the score obtained through the comparison module is greater than a threshold; otherwise, it is an imposter. A genuine comparison implies that the query biometric sample belongs to the claimed identity and the access is granted. A block diagram depicting the process of biometric verification is shown in Figure 1.3.

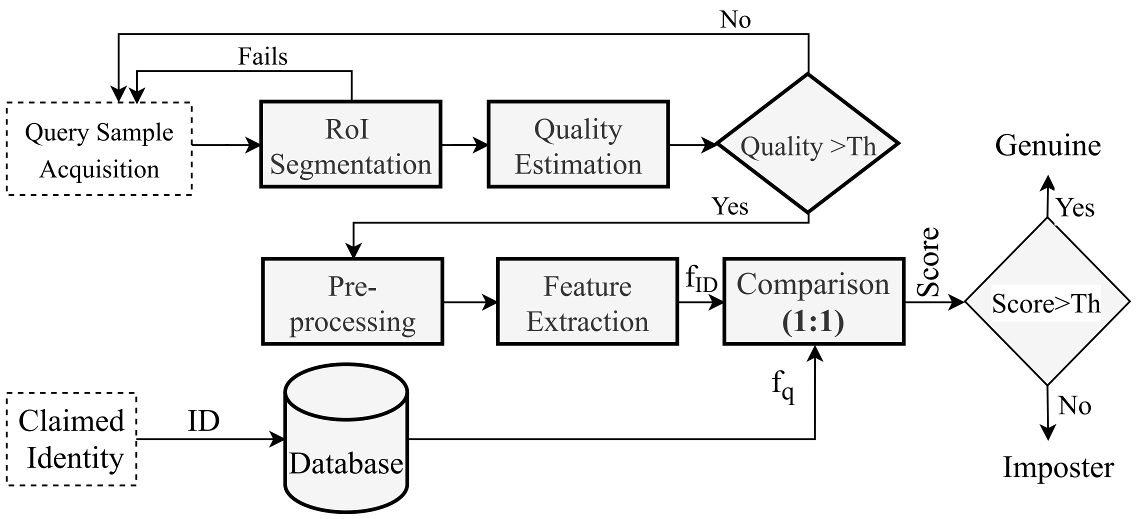


Figure 1.3: Flow chart of a biometric verification system

1.2.3 Identification

The objective of an identification system is to establish the identity of the newly acquired biometric sample, also known as query or probe biometric sample. In other words, it answers the question ‘whose biometric sample is this?’. This is useful in scenarios involving investigation regarding a face captured in the camera footage or latent fingerprint recorded from a crime scene. This can be accomplished by comparing the query sample with all the templates stored in the database and finding the most similar one. The query sample is pre-processed and its feature vector, denoted by f_q, is extracted using the feature extractor module. The query feature vector f_q is compared with all the templates stored in the database and a similarity score is computed for each comparison. The identification system outputs a list of k most-similar identities for the query sample. The performance of an identification system can be evaluated by determining the rank of actual identity from the list of identities. The block diagram of the identification process is shown in Figure 1.4. The query sample, denoted by f_q, is compared with all the templates or feature vectors in the database contained in the set F. Here, $F = \{f_{ID_1}, f_{ID_2}, \ldots, f_{ID_N}\}$ which further represents feature vectors corresponding to identity number $ID_1, ID_2, \ldots, ID_N$ respectively and N is the size of the database. The comparison module outputs a score list

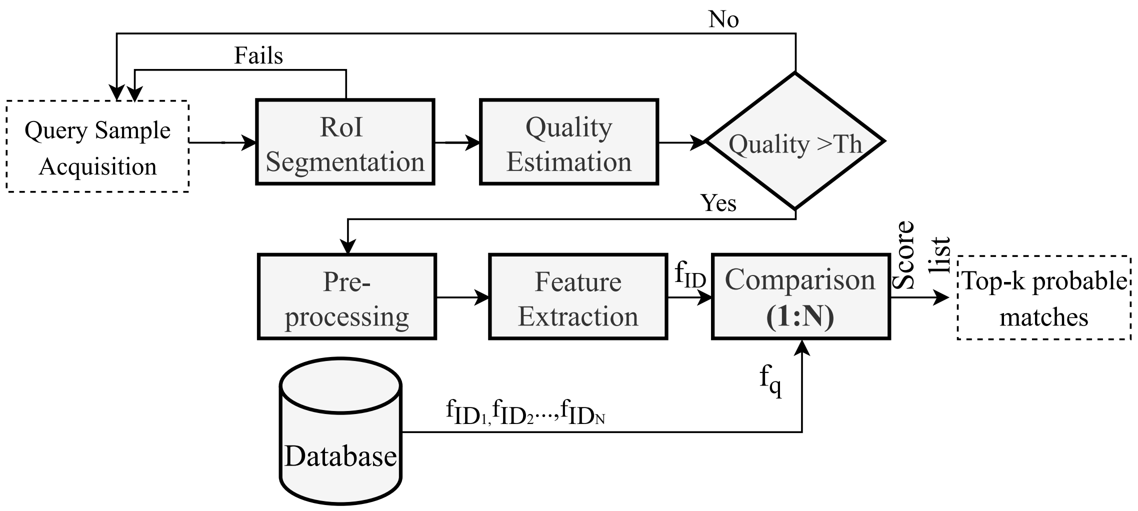


Figure 1.4: Block diagram of identification system

S, where $S = \{S_1, S_2, \ldots, S_N\}$. Here S_i represents similarity score when f_q is matched with a feature vector f_{ID_i} and $1 \leq i \leq N$.

1.3 Indexing of Biometric Databases

A biometric verification system involves only one comparison because the new biometric sample is compared with the database template, which is stored corresponding to the identity claimed by the user. On the other hand, an identification system compares the query sample with all the templates stored in the database to find its most probable match. Therefore, the number of comparisons required for the identification becomes proportional to N, where N is the number of templates stored in the database. Hence, the scalability of a large-scale identification becomes a challenge. There has been an increase in the number of enrollments with digitization and an increase in deployment of biometric authentication systems such as Aadhar in India [31] and MyKad in Malaysia [32]. This affects the throughput and error rate of an identification system. *Throughput* refers to the number of queries that the system can process in a particular period of time. *Error rate* is defined by the number of false positives that a system amounts to. Identification in large databases may suffer from poor throughput and a high error rate. Thus, there is a need to reduce the number of comparisons to find the true identity of the biometric query sample

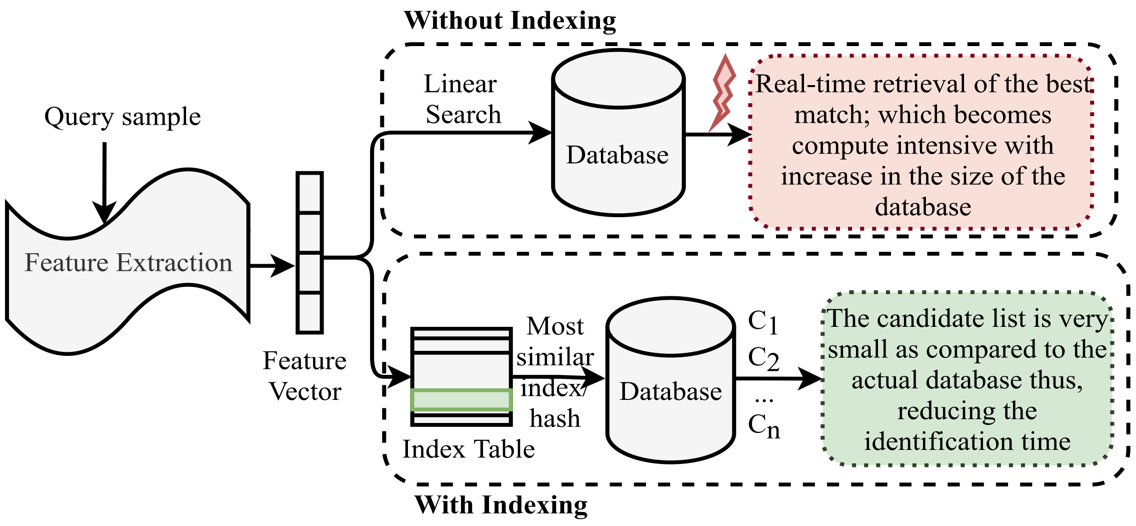

Without Indexing
Query sample
Feature Extraction
Feature Vector
Linear Search
Database
Real-time retrieval of the best match; which becomes compute intensive with increase in the size of the database
Most similar index/ hash
Index Table
Database
C1
C2
...
Cn
The candidate list is very small as compared to the actual database thus, reducing the identification time
With Indexing

Figure 1.5: Block diagram depicting the advantage of indexing a biometric database.

by narrowing down the search space. The technique that prunes the database and outputs a small list of candidates for comparison with the query sample to facilitate faster and efficient identification is referred to as *indexing* in this case. Indexing a biometric database eliminates the need of linear search in the database for finding the true match of the query sample. Rather, the index table returns only a subset of database, also called as candidate list $\{C_1, C_2, \ldots, C_n\}$ where $n \ll N$. Thereby, reducing the number of comparisons and accelerating the identification process. This is pictorially represented in Figure 1.5.

Indexing is generally carried out in three stages $viz.$ *feature extraction*, *index table generation* and *retrieval*. A feature extractor module is employed for learning feature vectors for biometric samples that represent its salient yet discriminating characteristics. During index table generation, each feature vector is associated with an index of the table. The similar feature vectors tend to lie in the same bucket of the index table. Therefore, the learned features should have high intra-class and low inter-class similarity so that the similar ones go in the same bin of the index table [33]. Both the stages mentioned above result in an indexed biometric database that is used for the identification of a biometric query sample. The objective of the retrieval stage is to output a list of candidates that can be used for comparison with the query sample to find its most probable match. So, whenever a biometric query sample is fed to the identification system that uses an indexed table, its feature vector is learned using the same feature extractor module. The extracted feature vector is compared with all the indices of the index table and the most similar index is found. The candidates lying in the bucket of the selected index are fetched for comparison with the query sample. The efficiency of a biometric indexing technique depends on the retrieval of a suitable candidate list from the index table, which further depends on the quality of the learned feature vectors. Therefore, the key idea is to find a suitable feature vector and generate an index table that could remarkably reduce the number of comparisons. Indexing biometric databases will narrow the search space thus, producing high throughput and reducing the error rate of an identification system.

1.3.1 Challenges in Indexing

There are certain challenges [34] that need to be accounted for while designing an indexing technique for a biometric database. These challenges appear due to different biometric sensors available for acquiring even the same modality or acquisition in an uncontrolled environment.

1. **High-dimensional features:** A biometric template is represented by a feature vector of $d-$dimension where d is large in dimension. Therefore, an appropriate data structure is required to store such large feature vectors. Multi-dimensional data structures such as $kd-$tree and $R-$tree are prone to the problem of curse of dimensionality. In other words, most of the nodes in these trees have to be traversed and thus, the search is no better than an exhaustive search.

2. **Variable number of features:** The extracted feature vectors for the biometric sample are real-valued and closely distributed in the feature vector space. It is also not necessary that the features extracted from a biometric sample at two different time stamps would be same. It may happen that some features could be missing while some false features could also appear. Hence, the features vary in number.

3. **No order:** There is no pre-defined order among the biometric features unlike some structural data such as text or numeric data. Therefore, they may appear in different

order which becomes a challenge.

4. **Occlusion and illumination:** Presence of occlusion and illumination variation in the acquired samples may affect the feature extraction process and thus, the indexing.

5. **Transformation:** The acquired image may be rotated, zoomed-in, zoomed-out etc. Therefore, the extracted features may vary even for the samples belonging to the same subjects.

6. **No Standardization:** Features extracted from each modality is different. For example, if minutiae and core-point is considered for fingerprints, they can't be used for iris. Hence, a generic indexing technique cannot be designed that works for all modalities databases.

1.4 Evaluation Parameters

Authentication of an individual requires comparing their learned feature vectors with the stored template(s) in the database. A one-vs-rest approach is followed to evaluate the performance of the learned feature vectors. Therefore, each biometric sample is compared with all the remaining samples, and a list comprising the similarity scores is generated. Comparison of two biometric samples is referred to as genuine if they belong to the same individuals. However, if the comparison is made between two samples belonging to different individuals, it is known as an imposter comparison. The authentication system gives an incorrect decision if the computed similarity score of a genuine comparison is below the threshold value or imposter comparison is above the threshold value. This scenario is called as *false rejection (FR)* and *false acceptance (FA)* respectively. Probability of the genuine and impostor score distribution for a biometric authentication system is shown in Figure 1.6. It can be observed that by varying the threshold, FA and FR changes and thus, the performance of a biometric recognition system depends on the threshold. A biometric authentication system designed for user convenience, such as those employed

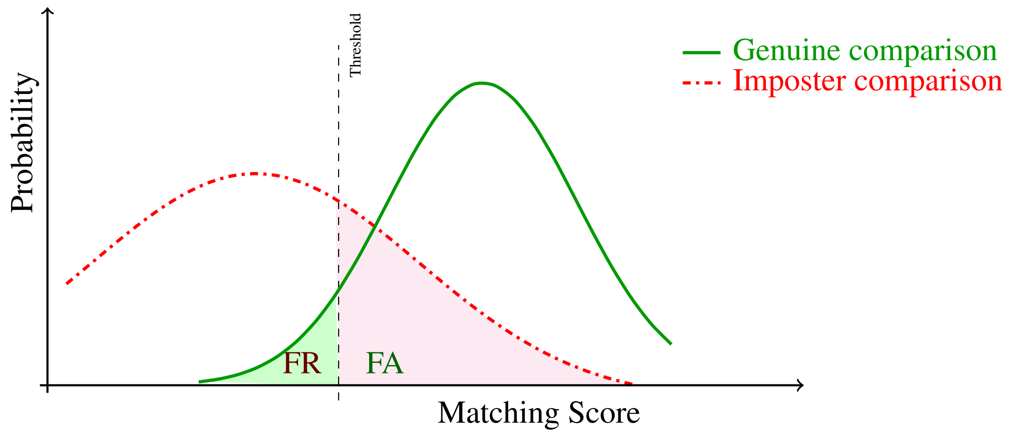


Figure 1.6: Graph depicting genuine and impostor score distribution for a biometric authentication system

in malls, can have a lower threshold value because it can afford a few genuine attempts to get rejected. However, a high threshold value is advisable when high security is required so that only a few imposter attempts are allowed by the system. The most common evaluation parameters for a biometric system are discussed as follows.

1. **False Acceptance Rate (FAR) and False Rejection Rate (FRR):** FAR is the percentage of the identification queries that got falsely accepted out of total number of samples shown to the biometric authentication system. FRR refers to the ratio of the genuine samples that got rejected by the system to the total number of identification queries made to the system. To understand, let us suppose there are four individuals $\{i_1, i_2, i_3, i_4\}$ are enrolled in a biometric database. An individual i_5 is trying to gain access to the system. Out of four attempts that he made, he gets identified as i_3 and i_1 in two attempts. Therefore, $FAR = \frac{2}{4} \times 100 = 50\%$. On the other hand, if i_1 is trying to gain access to the system and he gets rejected three times out of the four attempts that he made. Then $FRR = \frac{1}{4} \times 100 = 25\%$.

2. **Equal Error Rate:** FAR and FRR changes with the threshold and are inversely related with each other. One can get a point where FAR is equal to FRR by varying the threshold value. This is a point where the imposter and genuine sample cause equal amount of discomfort to the system. Therefore, a lower value of EER indicate

a better identification system.

3. **Decidability Index (DI):** It refers to the separation between genuine and imposter scores. DI is defined as given in the equation below, where $\mu_{(.)}$ and $\sigma_{(.)}$ denotes mean and standard deviation of either genuine or imposter scores.

$$DI = \frac{\mu_G - \mu_I}{\sigma_G^2 - \sigma_I^2} \tag{1.1}$$

4. **Accuracy:** Accuracy of a recognition system is defined as number of correctly recognized samples out of total queries made to the system. Mathematically it can be expressed as,

$$Accuracy = max\Big(100 - \frac{FAR + FRR}{2}\Big) \tag{1.2}$$

5. **Receiver Operating Characteristic (ROC) curve:** The ROC curve depicts the trade-off between false acceptance rate (FAR) and false reject rate (FRR) at different threshold values. It depicts the performance of a classification system at various classification threshold values in a comprehensive manner.

6. **Correct Recognition Rate:** CRR is defined as the rank-1 recognition rate $i.e.$ number of queries that have got their true match at the rank-1. If q_1 is the number of correctly recognized queries at rank-1 out of total Q queries made to the system, then CRR can be defined as given in (1.3)

$$CRR = \frac{q_1}{Q} \times 100 \tag{1.3}$$

7. **Hit Rate:** During identification, a query or probe image is referred to as correctly identified if one of the retrieved candidates belongs to the same identity as that of probe image. Therefore, Hit Rate (HR) is defined as the ratio of correctly identified queries (q) with respect to the total number of queries made to the system (Q), as

given in (1.4).

$$\text{Hit Rate} = \frac{q}{Q} \times 100 \tag{1.4}$$

8. **Penetration Rate:** It is the percentage of the database that needs to be retrieved for correct identification of the query image. If Q number of queries are made to the system and for each query i, C_i number of candidates are retrieved from a total of N templates, where N corresponds to the size of the database. Then, penetration rate (PR) can be defined as,

$$\text{Penetration Rate} = \frac{1}{Q}\sum_{i=1}^{Q}\frac{C_i}{N} \times 100. \tag{1.5}$$

9. **Cumulative Match Curve:** CMC is a rank-based metric that shows relationship between identification probability at a given rank. That is, what ratio of total queries got correctly identified till a particular rank.

A good biometric database indexing technique is expected to achieve high hit rate at low penetration rate. The relationship between these parameters can be understood from the graph shown in Figure 1.7. The graph shows relationship between hit rate and penetration rate for three hypothetical indexing techniques A, B and C. It is clearly evident that indexing technique A performs best among the three techniques. It is so because high hit rate is achievable at lower value of penetration rate.

1.5 Motivation

Biometric authentication is a widely used mechanism for access control. During identification it aims to determine identity of a biometric query sample by comparing it with all the samples in the database and using the most probable match. Identification is highly beneficial in scenarios requiring investigations. For instance, a fingerprint recovered from a crime scene or the suspect's face captured in CCTV cameras [7] must be identified in

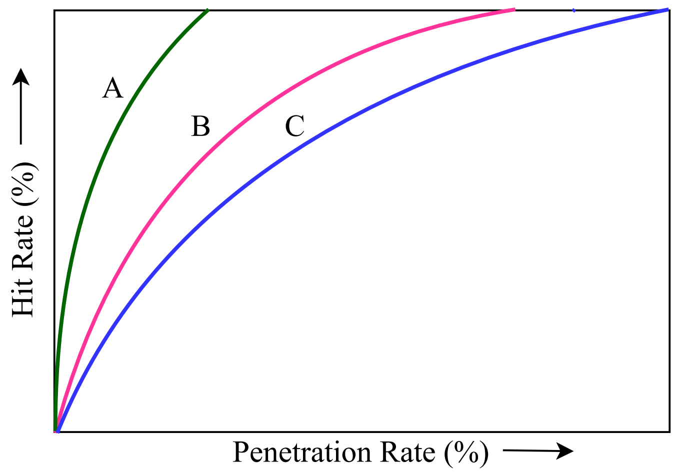


Figure 1.7: Graph showing relationship between hit rate and penetration rate for three biometric database indexing techniques.

order to get a lead. Identification is similar to finding nearest neighbors for a given biometric sample. This requires comparing the biometric sample with all the templates in the database using a similarity metric and finding the top-k matches like KNN [35]. The number of comparisons are proportional to the size of the database which makes this process computationally expensive with increase in the size of the database [36]. Identification can be accelerated by indexing the biometric databases to produce a small list of candidates for comparison against the query sample.

This thesis aims to fasten the process of biometric identification through indexing. Feature vectors play an important role in the performance of any indexing technique. For very long, hand-crafted feature extraction approaches such as SIFT [37], SURF [38], ORB [39], Local Binary Pattern (LBP) [40] are in use. However, these approaches try to maintain a trade-off between accuracy and computational efficiency [41] they are brittle for complex samples and large databases. Recently, a few automatic feature learning approaches have emerged involving deep learning (DL) that uses specialized neural networks such as Convolutional Neural Network (CNN) [42], Autoencoders [43], Siamese networks [44] etc. These approaches consider the complete images as input and learn discriminating and salient features of the sample. New approaches discover representations at multiple levels such that the learned feature vectors have high intra-class and low inter-class similarity [45]. These approaches instead of learning features from segmented images or using the developer's domain knowledge to construct hand-crafted features, consider raw images for the feature vector construction. Additionally, these approaches tend to improve with an increase in the number of training samples therefore, are more scalable and robust.

The research work in this thesis aims to develop novel deep learning based specialized indexing techniques for the biometric databases. We have considered four popular biometric traits $viz.$ iris, palmprint,finger-knuckle-print and fingerprint for the indexing. There are three components of our approach, feature extraction, index table generation, and retrieval. Specialized deep learning based network architectures are devised for learning feature vectors that are further used for indexing to facilitate faster identification. The architectures have been designed from scratch by keeping the targeted biometric modality in mind.

1.6 Research Gaps

Feature vector plays a significant role in the design of the index table and further, dictates the performance of the identification process. Handcrafted feature extraction approaches have been mainly used in the literature. These approaches tend to subside with increase in the size of the database as they try to maintain the trade-off between computational efficiency and accuracy. Moreover, the handcrafted features do not work well for biometric databases due to presence of large intra-class variance among the samples of the same subject. Advancements in deep learning have shown a way to automatically learn salient and discriminating features from the input samples. These techniques are data hungry and get better with increase in the size of the database. They are also capable to model complex non-linear structure from the high-dimensional data [46]. Although the deep learning models [42] proved to be efficient in many computer vision tasks but a limited attention has been paid to its applications in biometric database indexing. We found a research gap in obtaining deep features for indexing the biometric databases to accelerate the identification process. The proposed research work aims to design specialized deep neural networks for learning feature embeddings from the biometric databases of four modalities viz. iris, palmprint, finger-knuckleprint and fingerprint.

1.7 Research Objectives

The objectives of this research work are listed down as follows.

- The aim of this research work is to investigate the usefulness of deep features in indexing of biometric databases.
- Design specialized deep neural networks for learning discriminating feature embeddings for different biometric modalities in such a way that they have high intra-class and low inter-class similarity.
- Devise a corresponding retrieval strategy to produce a small and fixed sized candidate list for a given query such that the true match could be found in the retrieved list of candidates.
- This thesis investigates biometric databases of four modalities viz. iris, palmprint, finger-knuckleprint and fingerprint.

1.8 Thesis Contribution

This thesis proposes indexing techniques for biometric databases to facilitate a faster and efficient identification process. The main contribution of the thesis can be categorized based on the biometric database modality, which is indexed and used for the identification. Four modalities $viz.$ iris, palmprint, fingerprint, and finger-knuckleprint are considered in this work.

1. **Iris Database Indexing:** Iris is one of the most accurate biometric traits for human authentication due to its reliability, uniqueness, and stability. A novel indexing technique that targets an effective reduction of identification search space for the iris database is proposed in this thesis. A specialized convolutional neural network

architecture is designed, trained as a siamese network to construct compact feature vectors for iris images. The features are trained to have low inter-class and high intra-class similarity in the latent space representation. Extracted features are subsequently clustered using k-means and agglomerative clustering to generate an index table. Experiments have been conducted on widely used public iris databases $viz.$ CASIA Interval and CASIA Lamp.

2. **Palmprint Database Indexing:** A palmprint database indexing technique called PalmHashNet is proposed. It generates highly discriminative embeddings to create a fixed-size candidate list for comparison to make identification a constant time operation. Acquired palmprint images are fed to the feature extraction network, which is pre-trained using softmax loss. A margin is added to the softmax loss to minimize the intra-class distance between samples belonging to the same class. This ensures that the features have high intra-class and low inter-class similarity. k-means and locality sensitive hashing (LSH) is explored for index table creation. Experiments are conducted on four widely used palmprint databases $viz.$ CASIA, IITD-Touchless, Tongji-Contactless and Hong Kong Polytechnic University Palmprint II (PolyU II) database.

3. **Finger knuckleprint Database Indexing:** Finger-knuckle-print (FKP) is a skin pattern that appears on the backside of a finger around distal inter-phalangeal joints found over intermediate and third phalanges. An indexing technique for the FKP database, FKPIndexNet, is proposed to learn similarity-preserving hash codes for generating index tables. It proposes a specialized autoencoder network and a custom loss function to produce fixed dimensional feature embeddings used for indexing. The embeddings are regularized to ensure that they have high intra-class and low inter-class similarity. BallTree hashing, k-means clustering, and locality-sensitive hashing are investigated for index table creation. The proposed technique is tested on two publicly available FKP databases $viz.$, PolyU-FKP and IITD-FKP.

4. **Fingerprint Database Indexing:** A fingerprint database indexing is proposed that extracts softmax on Coaxial Gaussian Track Code (CGTC) for each fingerprint. CGTC is a fixed-length feature vector obtained for every minutia point based on its location in the fingerprint. This is followed by core point detection. The core point is the highest curvature point in the fingerprint. A stacked hourglass-based U-Net is proposed that outputs the coordinates of the core point in an input fingerprint. After getting the coordinates of the core point, its Euclidean distance from each minutia point is computed. This distance dictates the generation of the index table. The proposed technique is tested on FVC2004 database.

1.9 Thesis Organization

This thesis comprises of total six chapters. The following four chapters i.e. Chapter 2 to Chapter 5 address indexing of iris, palmprint, finger-knuckle-print and fingerprint databases respectively. Each chapter starts by introducing the respective biometric modality followed by a literature survey and then proposes a novel deep learning based method for indexing. Results are obtained on popular widely used open biometric databases and is compared with the state-of-the-art techniques.

Chapter 2 presents an indexing technique for iris databases. The proposed technique uses a specialized convolutional neural network architecture to learn a 1024-dimensional compact feature vector for iris images. k-means and agglomerative clustering are investigated for index table generation. The proposed technique is tested on two popular iris databases, CASIA Interval [47] and CASIA Lamp [48] and is found to achieve lower penetration rate as compared to the state-of-the-art techniques.

In **Chapter 3**, an efficient indexing technique for palmprint databases has been proposed. A metric-based deep learning network, called PalmHashNet, is proposed for feature extraction. It learns discriminative feature vectors of 512-dimension for a given palmprint images. An index table is generated by implementing k-means clustering and Locality-Sensitive Hashing (LSH) on the leaned feature vector space. The proposed technique is tested on four popular publicly available palmprint databases $viz.$ CASIA [49], IIT Delhi Touchless [50], Tongji Contactless [51] and PolyU II [52] palmprint dataset. All the databases contain palmprint images collected in an unconstrained environment. The results demonstrate the efficiency of the learned features in the identification process.

Chapter 4 proposes a novel approach for indexing the FKP database. A novel segmentation network for segmenting out the region of interest (RoI) from the acquired dorsal finger images is proposed. The extracted RoI is fed to the novel feature extraction network that outputs a 512-dimensional feature embedding for input FKP samples. This network uses a custom loss function to ensure high intra-class and low inter-class similarity among the learned feature vectors. Three techniques $viz.$ k-means clustering, Ball-Tree hashing, and locality sensitive hashing with nearest neighbor search are explored for index table construction. The proposed technique shows its efficacy when tested on two publicly available databases $viz.$, PolyU-FKP [53] and IIT Delhi Finger Knuckle Database [54].

Chapter 5 presents an efficient technique to index a fingerprint database. A novel autoencoder-based architecture with a stacked hourglass is proposed to detect the location of the core points from the fingerprint images. The minutiae points and the detected core point is used to create feature vectors such that they encode spatial and directional relationship between the core point and every minutia point. The feature vectors are used to generate a 2-dimensional index table. The proposed technique is tested on FVC2004 DB1_A database and was found to perform better than other techniques proposed in the literature.

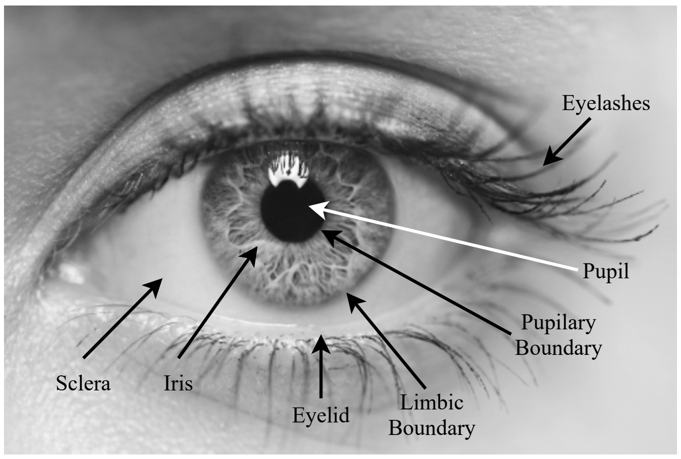
Lastly, the thesis is concluded by presenting the overall summary and future scope of this research work in **Chapter 6**.

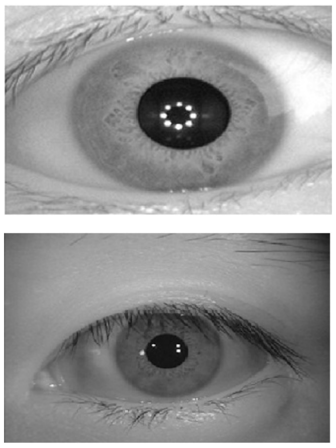
Chapter 2

IrisIndexNet: Indexing on Iris Databases

Iris is one the most accurate biometric traits for human authentication [2]. It is a ring of tissue observed around the pupil. The dilator and the sphincter muscles are responsible for controlling the size of the iris which further controls the amount of light entering the pupil [55]. A white region called sclera, that consists of connective tissues and blood vessels, surrounds the iris. The iris and pupil are covered by a clear layer called cornea. Figure 2.1 depicts the elements in the acquired iris sample. Iris contains rich pattern of ridges, furrows and pigment spots [56]. The minute details of the iris texture are developed during the fetal development of the eye. Therefore, these are believed to be unique between two individuals and also between different eyes of the same person. The iris patterns remain constant for most of a human's lifespan and thus, it is considered to be temporally invariant. Another advantage of using iris as a biometric trait is that is is an internally protected organ and therefore, it is difficult to forge [20]. It has a low false-matching rate as compared to other available biometric traits [19, 21].

Iris identification determines the identity of an input iris sample by comparing the probe iris image with all the templates stored in the database and finding the most similar





(a) (b)

Figure 2.1: Image depicting (a) elements in a human eye and (b) acquired iris samples.

one. Therefore, requiring N number of comparisons where N is the size of the database. Consequently, the time required to claim a person's identity becomes directly proportional to the size of the database. Iris indexing aims to make the identification process efficient by narrowing down the search space and reducing the number of comparisons. This expedites the identification process. An iris database indexing technique is expected to effectively prune the search space and provide probabilistic guarantees for a query to be identified in the selected bucket of the generated index table. The number of comparisons becomes bounded since the size of the bin is constant. The major challenge in the iris database indexing arises due to fuzziness among the samples. It may happen that the probe and gallery image belonging to the same subject may not be identical despite appearing similar. Another challenge is the trade-off between speed and accuracy.

The key contributions of this chapter is as follows. This chapter proposes a novel technique that aims at indexing the iris database to accelerate the identification process. It includes learning iris-specific features by employing a custom-designed siamese-based network. The proposed network is shallow and employs large-sized and more filters. Such an architecture design helps capture the variance present in different iris samples of the same subject while making the learned features corresponding to different class samples distant in the embedding space. It ensures learning of low-level textures effectively, which are the vital discriminating features for the iris images. The use of contrastive loss instead of triplet loss has further improved the captured variance by allowing us to train the network on more diverse comparisons through a larger sample size for comparable training time. The index table is generated using two techniques, k-means and agglomerative clustering, separately on two publicly available standard databases $viz.$ CASIA Interval and CASIA Lamp. The rest of the chapter is organized as follows. The next section gives an overview of the related work that has been done in iris database indexing. Section 2.2 describes the proposed technique in detail. Experimental setting and results are analyzed in Section 2.3.

2.1 Literature Survey

Many studies have made a significant contribution in investigating techniques for iris recognition, but there is a limited amount of work in iris database indexing. The pioneering work by Daugman et al. [19] uses beacon guided search utilizing multiple colliding segments. Mukherjee et al. proposed iris indexing technique using two different methods [57]. The first one uses Iris Codes generated from iris images. In contrast, the second one uses textural patterns in iris images using Signed Pixel Level Difference Histogram (SPLDH) of the raw pixel intensities. Gadde et al. [58] proposed an indexing technique by utilizing the clustering property of the Burrows-Wheeler Transform (BWT). The normalized iris image is initially converted to a binary image. A horizontal n-bit pattern is selected and the locations containing this pattern are searched for in the iris image. The image is segmented into vertical partitions and based on the presence of the selected pattern; it is assigned an index code. Another work by Mehrotra et al. [59] uses scale-invariant feature transform (SIFT) [37] and the generated keypoints are indexed using geometric hashing. Si et al. [60] proposed a technique for eyelash detection using directional filters. The paper further proposes an indexing technique by analyzing the texture information in the blocks of normalized iris images. This is done by detecting corners using Harris corner detection and analyzing the distribution of corners to get textural information. The index code is generated by dividing the normalized iris image into blocks and labeling each block. Jayaraman et al. proposed an iris indexing technique using both color and texture [61]. It uses color for index table generation and texture to find similar candidates to query images. The features are extracted using SURF and Kd-tree has been utilized to index high dimensional features obtained from color histograms. Dey et al. [6] proposed a 12-d index key generation from Gabour energy features. The paper extracts textural features from normalized iris images using a multiresolution Gabor filter applied across various scales and orientations. Drozdowski et al. [62] proposed an iris indexing technique using bloom filters and binary search trees. In this method, the iris image is divided into blocks and each block is passed through bloom filters. This results in a representation of the iris sample as a fixed-length sequence of bloom filters. Two samples are compared by computing the Hamming distance between the generated representation. A B-tree is created from all the enrolled samples and the search begins from the root of the tree. Khalaf et al. [63] divides the iris image into 8×8 blocks and extracts features from them by applying DWT, DCT and SVD. K-means++ algorithm is applied to the features for partitioning the embedding space. The search is carried out by dividing the database into two groups and creating two B-trees. Both the trees are searched, and the two most similar bins are retrieved for comparison. Another notable technique, proposed in [64], computes feature deviation that arises due to the presence of noise in the iris images. The feature deviation determines the search threshold for iris images based on which the index table is generated. Recently, Singh et al. [65] proposed patch level ordinal features for establishing relations between different iris images. The real-valued features are discretized and multi-index hashing is utilized for index table creation.

2.2 Proposed Technique

Iris is one of a reliable biometric modalities for pattern matching because of its highly distributed texture patterns. Nevertheless, due to obstructions like eyelids and eyelashes, it becomes difficult to detect the textural patterns present in the iris. However, it has been seen recently that Convolutional Neural Networks (CNN) [66] perform better in feature learning than the traditional handcrafted methods such as Gabor filter bank [67], Local

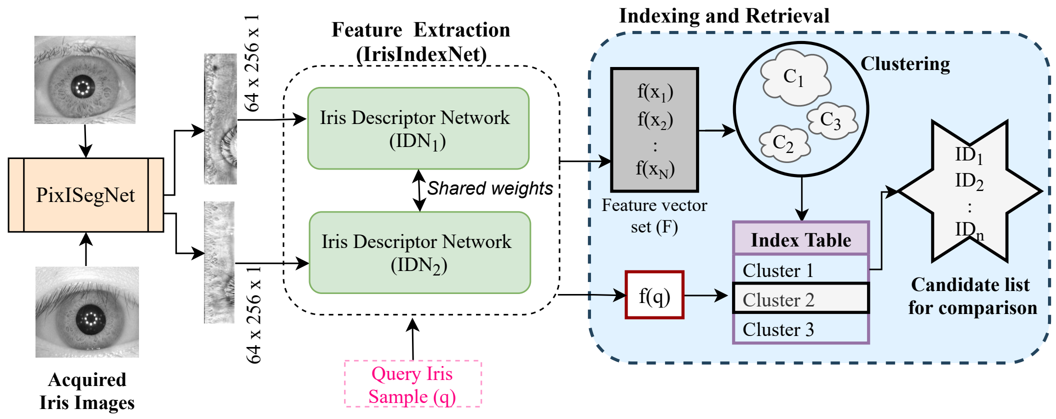


Figure 2.2: Block diagram of the proposed technique. The feature extraction module consists of training two identical networks as siamese architecture. The extracted features are clustered for index table generation.

Binary Pattern (LBP) [40], Laplacian of Gaussian (LoG) [68] etc. CNNs learn distinctive and salient features that best represent an iris image. The proposed technique learns a robust iris feature descriptor that aims to reduce the candidate list size for comparison to facilitate faster identification of a query iris sample. The proposed technique has four components, 1)RoI segmentation and Pre-processing, 2) feature extraction, 3) index table generation, and 4) retrieval. A block diagram of the proposed technique is shown in Figure 2.2. The raw iris images are subject to PixISegNet [20] for extracting region of interest. The RoI images are fed to feature extractor, IrisIndexNet, which consists of two similar Iris Descriptor Network (IDN_1 and IDN_2). After training, the feature vectors are extracted for the complete database of size N and the query iris sample (q). The feature vector set (F) is clustered for index table generation. Each index in the generated index table represent the cluster center and all the feature vectors lying in that cluster are put in the index's bucket. The query feature vector (f_q) is compared with all the indices and the candidates lying in the most similar index are retrieved for identification. All the modules are discussed in detail in the following subsections.

2.2.1 RoI Segmentation and Pre-processing

The iris samples may contain off-angles, specular reflection, motion blurs, noise, eyelashes, and eyelids when acquired in an unconstrained and non-cooperative environment

[69]. This dramatically affects the performance of an iris authentication system because the iris features are visible near the pupil boundary. Therefore, an accurate iris region of interest (RoI) segmentation is required for better feature extraction and, thus, iris identification [70].

In this work, iris images are initially pre-processed to segment out the region of interest (RoI) containing iris pattern using the technique proposed in [20]. This technique uses a U-Net architecture [71] that consists of a contacting and an expansive path. The contracting path consists of a series of convolutional layers followed by the max-pool layer. On the other hand, the expansive path takes the output of the contracting path and puts it through a series of transposed convolutional and up-sampling layers. The high-level features from the contracting path are concatenated with the corresponding up-sampled features in the decoder using merge connections. U-Net is commonly used in applications involving the segmentation of medical images. The observed output was irregular and blurry when the standard U-Net was used for segmentation RoI from iris images. It was due to occlusion, pose and scale variation that gets introduced in the iris images during acquisition in an uncontrolled environment. This limitation is addressed by introducing a stacked hourglass network in between the contracting and expansive path. Hourglass network [72] includes a residual module that works on the feature vector, unlike the U-Net, which takes a complete image as an input. The residual module uses convolutional operation to learn high-level features, but it is also capable of retaining the original information with the use of skip connections. It has a symmetric topology so that the features are extracted and consolidated across various image scales and resolutions. The output of the hourglass network is of the same size as that of its input feature vector. Therefore, we can say that the hourglass network only changes the depth of the data without altering its size.

The iris RoI segmentation network has three components $viz.$, contracting path, expansive path and the hourglass network. The contracting path aims to learn salient and discriminating features corresponding to the input iris images. The contracting path consists of a pair of convolutional layers with a filter size of 3×3 and stride of 1, followed

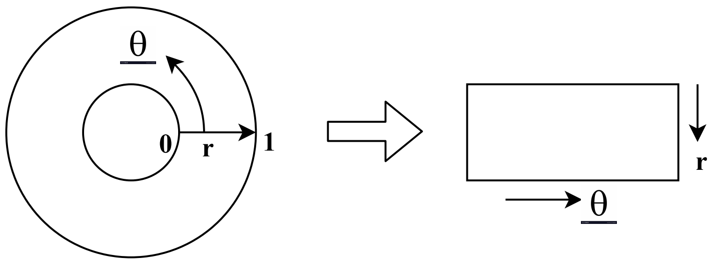


Figure 2.3: Iris Normalization procedure.

by ReLU activation and lastly, a maxpool layer with size = 2×2 and stride of two. This block is repeated three times with different filter sizes of 16, 32, and 64. Unlike the standard U-Net, three hourglass networks are stacked over each other and are introduced in the bottleneck of U-Net for better RoI segmentation. Therefore, the output of contacting path is passed to the hourglass network. Each hourglass consists of an encoder and decoder that has four residual modules connected in a sequential manner. After passing the feature vector through three hourglass networks, the output is given to the expansive path. The expansive path consists of a pair of transposed convolution layers with filter size of 3×3 followed by an up-sampling layer. This block is repeated three times followed by a 1×1 convolutional layer. The output of each block is concatenated with the corresponding feature map in the contracting path using a merge connection. The output of 1×1 layer in the expansive path is the segmented region of interest (RoI) containing the iris image.

The segmented iris images are normalized. The images are transformed from normal coordinate system to polar coordinate system. It makes the iris images of a fixed dimension which further allow comparison of the images. The inconsistency between the same subject's iris image arise due to varying size of the iris due to pupil dilation. The mapping of iris image to polar coordinates (r, θ) is achieved by using Daugman's rubber sheet model [73]. As per the model, r moves in distance range of $[0, 1]$ and θ moves in the angular range of $0, 2\pi$, as shown in Figure 2.3. Mapping to polar coordinates is achieved by,

$$I(x(r,\theta), y(r,\theta)) = I(r,\theta) \tag{2.1}$$

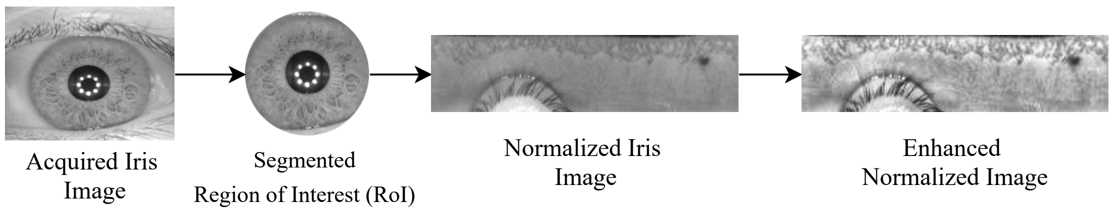


Figure 2.4: Figure showing Region of Interest Segmentation and normalization of an acquired iris image taken from CASIA Interval database.

$$\begin{aligned} x(r,\theta) &= (1-r)x_p(\theta) + rx_l(\theta) \\ y(r,\theta) &= (1-r)y_p(\theta) + ry_l(\theta) \end{aligned} \tag{2.2}$$

Here, the coordinates of the pupil and iris boundaries along the θ direction are x_p, y_p and x_l, y_l. This model is robust to pupil dilation and size inconsistencies. The stages of iris segmentation and normalization with respect to an acquired iris image are shown in Figure 2.4.

2.2.2 Feature Extraction

The normalized images are fed to the proposed feature extraction network, called IrisIndexNet, to learn a compressed representation for an iris image. The key idea is to learn a feature representation function that maps similar iris images $i.e$ the ones belonging to the same subject closer in the feature-embedding space. While, the images belonging to different subjects should have dis-similar feature vectors and should be far from each other in the feature embedding space.

IrisIndexNet is a siamese network [44] based architecture. Siamese network is a category of neural networks that consists of two identical networks. Two networks are identical if they have the same configuration, parameters, and weights. The parameters are updated for both networks simultaneously. Different images are fed to the networks that learn feature vectors corresponding to them. The networks are trained to minimize the distance between feature vectors belonging to images of the same subjects, and the distance between feature vectors of different images is maximized in the feature embedding space. The weights are updated for both networks simultaneously. In other words,

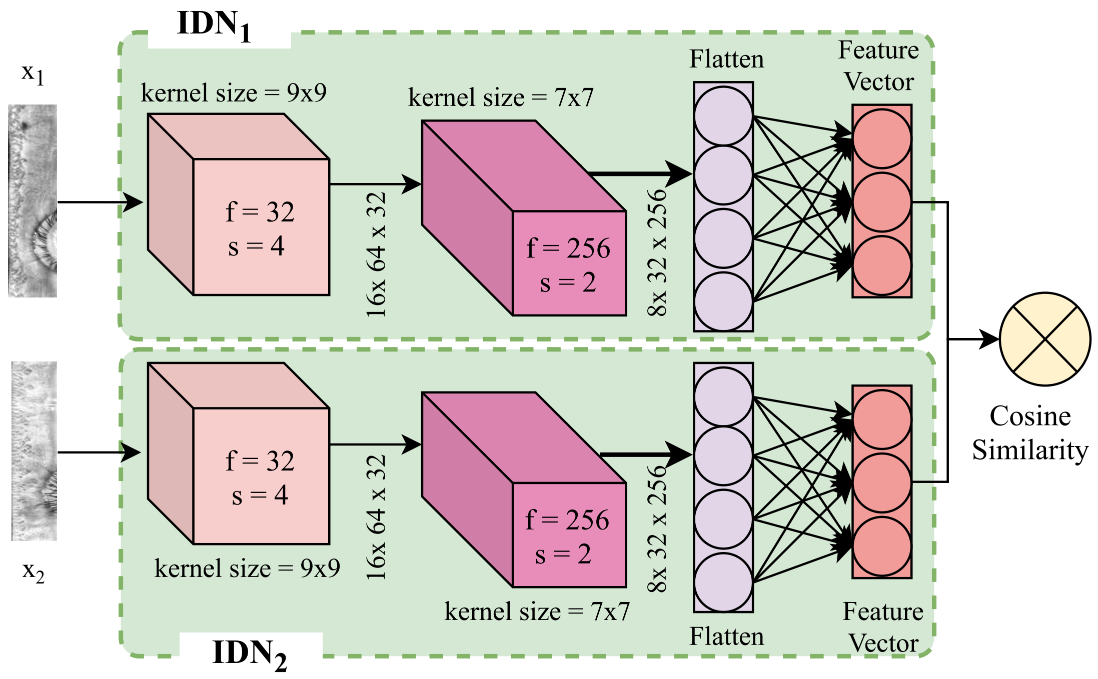


Figure 2.5: Architecture diagram of the proposed IrisIndexNet consisting of two iris Descriptor Networks (IDN_1 and IDN_2).

siamese networks learn a similarity function that outputs if two input images are similar or not. When two images belong to the same subject, it is referred to as a positive pair. While a negative pair refers to two images belonging to different subjects. Training a siamese network poses an advantage if there is a class imbalance in the training dataset. Class imbalance refers to when the number of positive pairs is quite less than the number of negative pairs or vice versa.

Iris Descriptor Network (IDN): IrisIndexNet is a feature extraction network that has two branches of the replicated Iris Descriptor Network (IDN), IDN_1 and IDN_2. The feature vectors obtained from IDN for two different iris images are compared by computing cosine similarity between the two. IDN maps an input image x_i to a latent representation $f(x_i)$ as given,

$$f(x_i) = IDN(x_i) \quad = \quad \phi(W_2^{IDN} \circledast (\phi(W_1^{IDN} \circledast x_i))) \tag{2.3}$$

where, W_1^{IDN} and W_2^{IDN} are the weights of the filters at the first and second layer, respectively. The operator $\circledast$ refers to the convolution operation. x_i is the input image vector, and ϕ denotes the ReLU activation function applied to introduce non-linearity on the output of both layers. IDN architecture consists of convolution layers followed by a flattening and dense layer that outputs a feature vector corresponding to the input iris image. The input image is passed through convolution layers having 32 and 256 filters with a kernel size of 9×9 and 7×7 and stride length of 4 and 2 respectively. It was observed that adding a max-pool layer after convolution layers resulted in the loss of the captured texture information. Therefore, strides of 4 and 2 have been used in convolutional layers as an alternative. These strides are compensated by the large size of filters $i.e.$ 9×9 and 7×7, which helps capture patterns of a larger locality. Lastly, the number of channels is increased from 32 to 256 from the first to the second layer as there was a huge amount of variance in the iris images, which was better captured using a large number of filters. These are followed by feature flattening and a fully connected dense layer. The fully connected layer acts as the feature embedding of size 1024 dimension. Cosine similarity is computed between the feature vectors coming from IDN_1 and IDN_2. The output value lies in the range of $[-1, 1]$, but the network outputs an absolute value for training the IDN. The architecture of the IrisIndexNet consisting of two IDNs is shown in Figure 2.5. In the figure, x_1 and x_2 refer to two normalized iris images.

2.2.3 Indexing

The objective of this phase is to associate each feature vector generated by IrisIndexNet, to an index. The output of IrisIndexNet is a feature vector set F comprising of feature vectors $\{f(x_1), f(x_2), \ldots, f(x_N)\}$ of all the input iris images x_i where, $i = 1, 2, \ldots, N$. Two types of clustering techniques namely, agglomerative [74] and k-means [75] are implemented on the feature vector set F. Class impurity and size deviation are utilized to evaluate the clustering quality. Class impurity linearly depends on the number of clusters in which the samples of a particular class are split in. Size deviation refers to the standard deviation in the distribution size of the clusters.

Algorithm 1 Indexing the extracted feature vectors

```
Require: Embedded feature vector set $F$
Ensure: Index table $IT$
1: for every iris image $x_i$, $i \in \{1, 2, \ldots, N\}$ do
2:     Extract the feature vector, $f(x_i)$ using IrisIndexNet
3:     Append $f(x_i)$ to $F$
4: Apply k-means or Agglomerative clustering on feature embedding space $F$ till convergence
5: for each cluster center or hash index $h_j$, $j \in \{1, 2, \ldots, k\}$ do
6:     Create an index with $h_j$ in the index table $IT$.
7:     Put the members of the cluster or partition in the bucket corresponding to $h_j$
8: return $IT$
```

Agglomerative Clustering [74]: As the name suggests, agglomerative clustering refer to aggregation or merging the clusters at various levels based on a similarity metric [76]. It is a bottom up approach which means that initially, each data point is treated as a different cluster. The similar clusters are later merged by computing a distance metric between them and this process continues till all the data points are grouped together in a single cluster. There are different types of linkages that describe the different approaches to measure the distance between two clusters. These are single, complete, average and ward linkage. Suppose there are two clusters C_1 and C_2. The single and complete linkage outputs the minimum and maximum distance respectively between two data points p_1 and p_2 such that $p_1 \in C_1$ and $p_2 \in C_2$. The average linkage between two clusters, C_1 and C_2, computes the average distance between all the pair of points contained in both the clusters. In this work, the affinity between clusters is calculated using cosine similarity for the above mentioned linkages [77]. On the other hand, Euclidean distance is employed for Ward linkage [78]. Class impurity and size deviation are computed for all the linkages at different distance threshold values as shown in Table 2.1. It was observed that complete linkage produces similar sized clusters for most threshold values without much reduction of class impurity. Whereas the single linkage quickly merges nearby clusters causing it to get to zero impurity at the cost of skewed cluster distributions. The average linkage strikes a balance in merging of clusters as the size deviation does not increase abruptly while the class impurity reduces consistently. On the other hand, ward linkage method

Table 2.1: Agglomerative clustering trials using absolute cosine similarity for CASIA Interval and CASIA Lamp dataset.

D.T.	**Complete Linkage**		**Single Linkage**		**Average Linkage**	
	S.D.	*C.I.*	*S.D.*	*C.I.*	*S.D.*	*C.I.*
CASIA Interval						
0.2	1.63	1.46	2.29	0.91	1.99	1.18
0.4	2.04	0.36	192.58	0.01	2.20	0.21
0.6	3.06	0.24	-	-	16.93	0.13
0.8	10.33	0.22	-	-	623.69	0.01
CASIA Lamp						
0.2	3.22	5.54	5.17	3.69	4.26	4.66
0.4	5.36	3.99	7.29	1.05	6.68	1.52
0.6	7.67	2.05	10.52	0.20	6.58	0.49
0.8	3454.15	1.26	4100.91	0.00	6.28	0.11

Table 2.2: Agglomerative clustering trials using Euclidean distance with ward linkage.

D.T.	**0.2**	**0.6**	**1.0**	**1.4**	**1.8**	**2.0**	**2.2**
CASIA Interval							
S.D.	0.07	1.42	1.68	1.83	3.55	**4.56**	6.94
C.I.	4.43	1.78	0.30	0.17	0.14	**0.14**	0.13
CASIA Lamp							
D.T.	**0.2**	**0.6**	**1.0**	**1.4**	**1.8**	**2.0**	**2.2**
S.D.	0.009	2.01	5.51	5.32	**3.76**	3.60	3.25
C.I.	14.99	6.89	1.64	0.43	**0.19**	0.14	0.12

gave better result. The value of size deviation and class impurity is optimal at 2.0 and 1.8 for CASIA Interval and CASIA Lamp database respectively, which is better than other linkage types. It is shown in Table 2.2. Euclidean distance threshold of 2.0 and 1.8 are chosen that gives 229 and 881 clusters for CASIA Interval and Lamp respectively. In both the tables Table 2.1 and Table 2.2, D.T., S.D. and C.I. refer to distance threshold, size deviation and class impurity respectively.

k-means Clustering [75]: The objective of k-means clustering algorithm is to partition a set of feature vectors F into 'k' disjoint groups. Each group or cluster has a representative data point also known as mean of all the feature vectors. The algorithm starts by initializing 'k' centers using k-means++ initialization. Euclidean distance of each feature

Table 2.3: Size deviation and class impurity with k-Means Clustering for different number of clusters for CASIA Interval dataset.

Clusters	**3**	**30**	**60**	**90**	**99**	**111**
S.D.	67.72	26.92	20.03	15.53	12.68	**12.34**
C.I.	0.66	0.83	0.71	0.61	0.61	**0.59**

Table 2.4: Size deviation and class impurity with k-Means Clustering for different number of clusters for CASIA Lamp dataset.

Clusters	**3**	**15**	**30**	**60**	**70**	**78**
S.D.	560	248.01	213.14	137.07	126.60	**101.39**
C.I.	0.52	0.63	0.67	0.60	0.62	**0.58**

vector is computed from all the centers and it is assigned to a cluster with least Euclidean distance. The cluster centers are updated after every iteration. It is found out by computing the mean of all the candidates assigned to that cluster. This is repeated till no further change in cluster assignment is observed or maximum number of iterations have been exhausted.

Class impurity and size deviation are computed for different values of k to determine appropriate number of clusters. It was observed that size deviation was considerably higher in k-means clustering as compared to agglomerative clustering and the class impurity saturated at relatively higher values. k-means was found to perform optimally at k values of 111 and 78 for CASIA Interval and CASIA Lamp dataset respectively, before the metrics saturated. The same is shown in Table 2.3 and Table 2.4. Cluster labels determined by these techniques need to be trained on a classifier before it can be used for predictions. In this work, Support Vector Machine is used. The computed clusters serve as class labels for training an SVM classifier for the feature vectors. Since the feature space is representative and has a large training data, a linear kernel is selected for the SVM classifier.

2.2.4 Retrieval

During the retrieval phase, the probe iris image (q_j), where $j = \{1, 2, \dots, m\}$ is fed to the trained Iris Descriptor Network for feature vector extraction. This feature vector is

Algorithm 2 Retrieval (IT, q)

```
Require: $IT$: Index table, $q$: probe iris image
1: Extract the latent representation, $f(q)$
2: for every index $ind \in IT$ do
3:     Determine the cosine similarity between $ind$ and $f(q)$
4:     Find the most similar index
5:     Retrieve candidates from the most similar index
6: return Candidate list
```

then passed through the classifier to determine the label for the most similar cluster. The candidates lying in the selected cluster would be used for comparison with the probe iris image, q_j using cosine similarity. A score list S is created which consists similarity score of a query feature vector with the candidates lying in the selected index of the index table. The list is used to find the rank of the q_j's true match. This is repeated for every probe iris image. The proposed technique reduces the search space by a huge margin thus, implementing a constant time search complexity for identification process. The retrieval process is explained through Algorithm 2.

2.3 Experimental Results

This section demonstrates the recognition as well as the indexing performance of the proposed technique on two standard iris databases. The details regarding the databases and the results obtained is given in the subsections to follow.

2.3.1 Datasets

The proposed model is tested on two publicly available standard datasets $viz.$, CASIA Interval [47] and CASIA Lamp [79]. The datasets are split in two different manners $i.e.$ horizontally and vertically. Horizontal split refers to when all samples of the selected subjects are used to train feature extraction. Vertical split is done by selecting the fixed number of samples from all the subjects to train the clustering module for indexing. Since there is no standard training and testing protocol associated with the considered datasets; both are partitioned in 80-20% train and test split. In both the datasets, 80% of the samples are used as gallery images while the remaining 20% are treated as probe iris

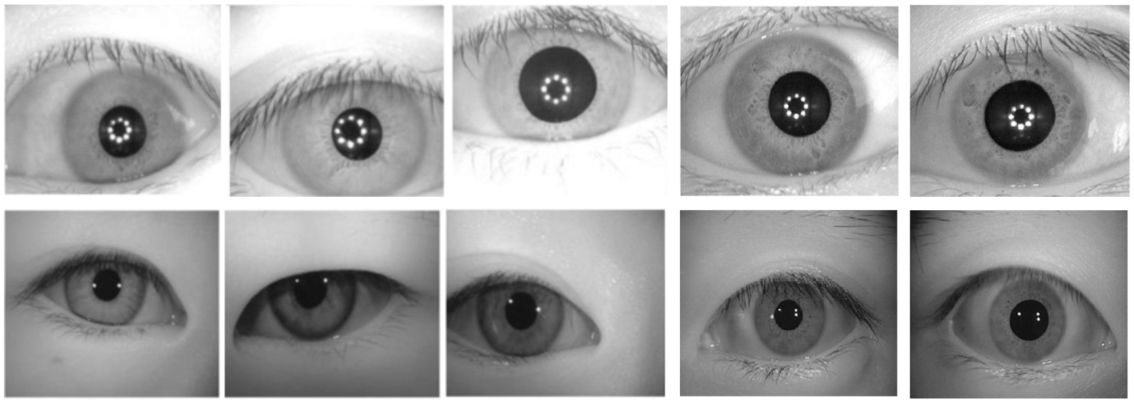


Figure 2.6: Iris Samples Images from CASIA Interval and CASIA Lamp Database in first and second row respectively.

samples. Some iris samples from both the datasets are shown in Figure 2.6.

CASIA Interval [47]: There are 2555 iris collected from 349 subjects. The images were acquired from a special close-up iris camera and have a resolution of 320×280 pixels. The images are good for extracting textural patterns.

CASIA Lamp [79]: CASIA Lamp is a larger dataset with 15,660 iris images collected from 819 subjects using a hand-held iris sensor. During acquisition, a lamp is kept near the subject and it was switched on and off in order to introduce more variations in the samples of the same subject. Due to illumination variation, pupil tends to expand and contract and the images in this database capture that. Thereby, making it a good database to test the robustness of extracted features. The images are collected in single session and have a resolution of 640×480 pixels.

2.3.2 Training and Testing Protocol

The Siamese network replicates the IDN and generates two feature vectors. The network generally needs to be trained for all $N \times N$ combinations of images and should output a binary value. The value is one if the input patterns belong to the same subject and zero in case the images are of different subjects. But, the number of positive pairs are quite less than the number of negative pairs. Due to this massive class imbalance there could be an increase in the number of false positives and this may lead to poor training of IrisIndexNet. To address this problem, the proposed technique samples out some random r

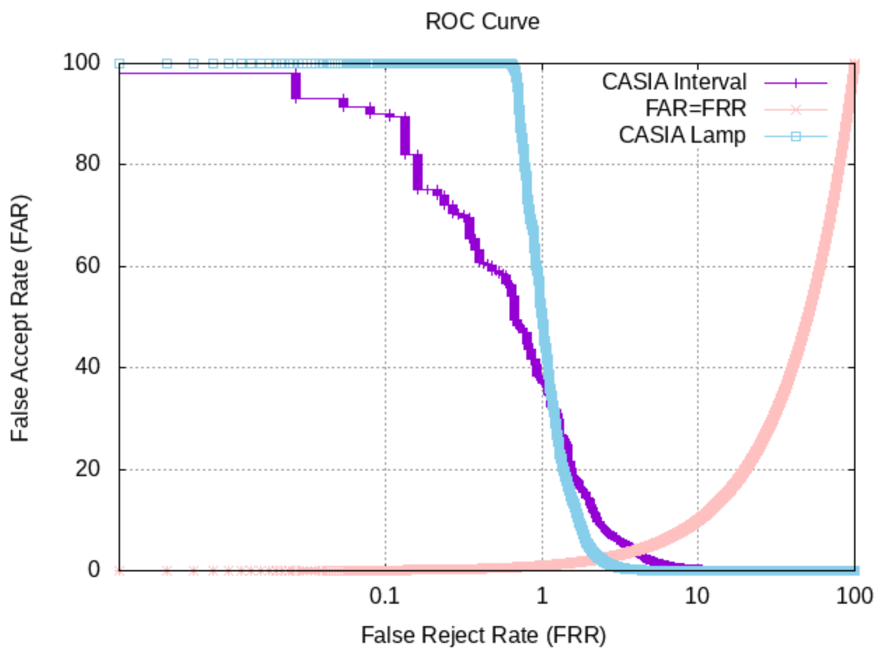


Figure 2.7: Receiver Operating Characteristic Curve, plotted on the log scale, for the considered databases.

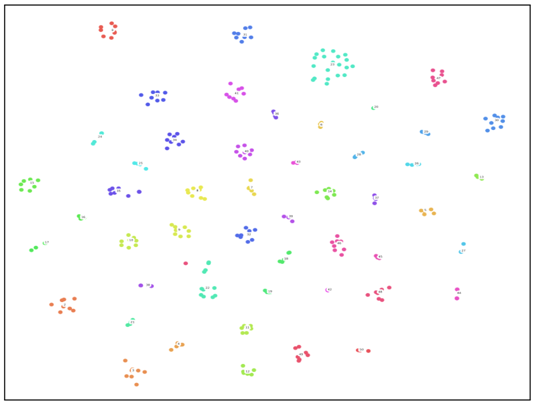
images of the same subject and l images of other subjects resulting in $(r + l) \times N$ combinations for training. The network is trained using a binary cross-entropy loss function as the output produced from the network lies between [0, 1]. Adam optimizer with a steady learning rate of 0.00006 has been used for training with a mini-batch of size 40. Weights of the convolutional layers have been initialized as per the zero-mean Xavier normal initialization. The training produces two gradients corresponding to each input pattern on the same weights because the same IDN has been used to produce the feature vectors for two images.

2.3.3 Recognition performance

The feature vectors generated using the proposed IrisIndexNet are tested for their recognition performance to validate the objective of obtaining 'high intra-class and low inter-class similarity'. Recognition performance is evaluated in terms of correct recognition rate (CRR), equal error rate (EER), discriminative index (DI) [34] and Accuracy. To compute these parameters, each test image is matched with all the images in the training

Table 2.5: Recognition performance of the proposed approach

Parameter	**CASIA Interval** [47]	**CASIA Lamp** [79]
CRR	96.45%	99.64%
EER	3.90%	2.36%
DI	2.63	2.69
Accuracy	96.55%	98.12%


(a) t-SNE CASIA Interval

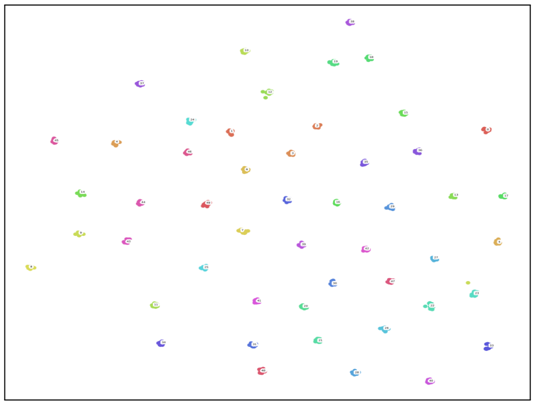

(b) t-SNE CASIA Lamp

Figure 2.8: t-SNE plot showing 50 randomly chosen classes for (a) CASIA Interval and (b) CASIA Lamp dataset

partition and similarity score is obtained.

Two types of comparisons are possible, viz. Genuine and Imposter. For CASIA Interval dataset, the number of genuine comparisons were 3739 while the imposter comparisons were 1231997. The number of genuine and imposter comparisons for CASIA Lamp dataset were 50122 and 39183739 respectively. Receiver Operating Characteristic Curve (RoC) is plotted for both the datasets Figure 4.7. It depicts the classification performance at various thresholds. Area Under the Curve (AUC) refers to the degree of separability $i.e$ it depicts the capability of the model in distinguishing between classes. Higher AUC indicate better model. The proposed approach achieved AUC of 0.9999 and and 0.9998 in CASIA Interval and CASIA Lamp Database.

The proposed technique achieved CRR = 96.49% and 99.64% and EER = 3.90% and 2.36% on CASIA Interval and CASIA Lamp dataset respectively. The same has been tabulated in Table 2.5. The learned features can be visualized using t-SNE plots. t-SNE

Table 2.6: Penetration rates(%) corresponding to various values of hit rate for CASIA Interval and CASIA Lamp database

	Agglomerative		**k-means**	
Hit Rate (%)	**Interval**	**Lamp**	**Interval**	**Lamp**
95	-	-	0.104	-
96	0.052	-	0.681	-
97	0.104	-	2.097	-
98	0.734	-	3.775	0.008
99	2.254	0.008	7.236	0.255

is a probabilistic technique for dimensionality reduction and is well suited for the visualization of high-dimensional feature vectors in a 2-D plot. Figure 2.8 shows t-SNE plots [80] for the learned features from CASIA Interval and CASIA Lamp database. Every color in t-SNE plot represents a class or subjects. It is clearly evident that the same class features are close to each other in the feature representation space while different ones are farther. This validates the performance of the extracted features using the proposed technique. Although, recognition performance is not the main aim of this study but the proposed technique is able to achieve good results thus, indicating the high discriminating and representative ability of the learned feature vectors.

2.3.4 Indexing performance

The learned feature vectors are clustered using k-means and Agglomerative clustering to generate an index table. Whenever a query iris image is given to the iris identification system, its feature vector is extracted using the trained IrisIndexNet. It is then compared with all the indices and the candidates lying in the most similar index are retrieved for comparison. The proposed technique achieves a hit rate of 99% at 7.236% and 0.255% penetration rate in CASIA Interval and CASIA Lamp dataset respectively. However, the penetration rate further reduced to 2.254% and 0.008% respectively for the considered datasets when Agglomerative clustering is applied. A 100% hit rate was achieved at 24.383% and 38.210% for CASIA Interval and CASIA Lamp respectively in case of k-means clustering. On the other hand, the penetration rate was 19.507% and 3.248% respectively when agglomerative clustering is applied. The penetration rate achieved by

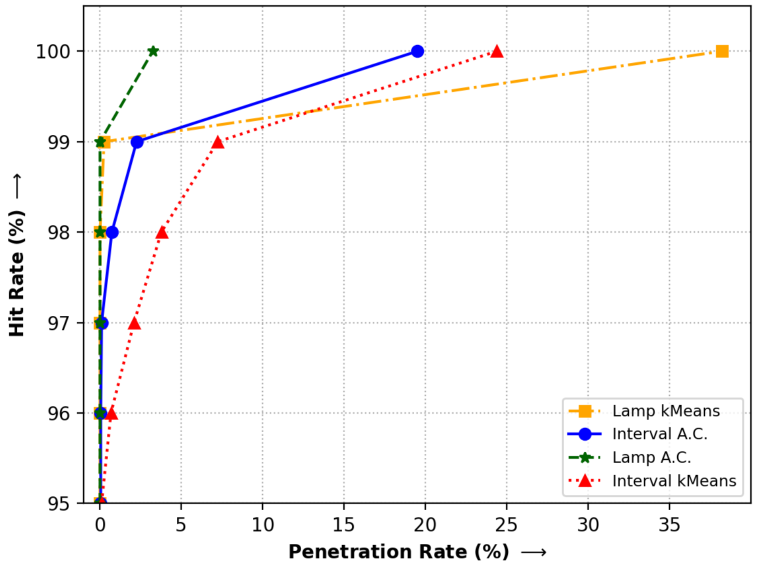


Figure 2.9: Hit rate vs. penetration rate on the considered datasets when agglomerative (A.C.) and k-means clustering has been used for indexing.

the proposed technique corresponding to different values of hit rates is shown in Table 2.6. The same is graphically represented in Figure 2.9. It can be observed that agglomerative clustering performs better than k-means clustering. This can be justified as agglomerative clustering defines clusters based on distance thresholds and thus, produces uniformly sized clusters.

2.3.5 Time analysis

The time-based performance of the proposed technique is evaluated in terms of speed-up. It refers to how fast the identification process has become by using the proposed technique for indexing as compared to the naive approach for identification that uses linear scan in the database. Speedup is calculated by the time taken to find a suitable candidate set from the created index table for matching with the query image. Table 2.7 shows a comparison of the query time taken for iris identification with and without indexing for both the types of clustering techniques on the considered datasets. The time taken by the query set is shown in seconds for both the datasets when k-means and agglomerative

Table 2.7: Identification time analysis of the proposed approach with non-indexed approach. The cell values show the time taken (in seconds) by the test set in both the datasets by k-means and agglomerative clustering.

Database	Without Indexing	k-means	A.C.
CASIA Interval	138.04	33.748	33.378
CASIA Lamp	35416.65	1280.711	1303.08

Table 2.8: Comparison with existing state-of-the-art techniques.

Technique	Features	HR(%)	PR(%)
Database: CASIA Lamp [79]			
Singh [65]	POFNet	99.82	0.62
Proposed	**IrisIndexNet**	**99.90**	**0.03**
Database: CASIA Interval [47]			
Mukherjee [57]	Iris texture & Iris codes	80.00	8.00
Gadde [58]	BWT	99.83	17.23
Dey [6]	Gabor features	91.1	14.5
Drozdowski [62]	1D Log Gabor	98.00	10.0
Khalaf [63]	DCT, DWT & SVD	69.63	0.98
Ahmed [64]	Deviation Features	98.77	3.40
Singh [65]	POFNet	98.73	2.11
Proposed	**IrisIndexNet**	**98.73**	**1.17**

clustering are applied for indexing the database prior to identification. It can be observed that the speedup is around 4 and 27 times in CASIA Interval and CASIA Lamp dataset respectively, if the proposed technique is applied for indexing the iris database.

2.3.6 Indexing Performance Comparative Analysis

The proposed technique is compared with the state-of-the-art iris indexing techniques for CASIA Interval and CASIA Lamp database. Effectiveness of the proposed technique can be seen from Table 2.8. The proposed technique achieves better penetration rate of 0.03% at a hit rate of 99.90% on CASIA Lamp database. Whereas, on the same database, the state-of-the-art technique proposed by Singh et al. [65] achieves a comparatively higher penetration rate of 0.62% at 99.82% hit rate. Table 2.8 also compares the proposed technique with state-of-the-art techniques with respect to the CASIA Interval database. Gadde et al. [58] uses clustering property of Burrows-Wheeler Transform (BWT) and achieves a high hit rate of 99.83% on CASIA Interval dataset but at a higher penetration rate of 17.83%. The recently proposed techniques in [64] and [65] achieves a hit rate of 98.77% and 98.73% at 3.40% and 2.11% respectively. The proposed technique achieves a hit rate of 99% at 2.25% penetration rate. For comparison with [65], the penetration rate of the proposed technique for 98.73% hit rate is computed and it came out to be 1.17%. Lower penetration is expected from an efficient indexing technique. Therefore, the proposed technique outperforms the techniques proposed in literature for the considered databases. It can also be seen that the there is significant reduction in the search space for searching the true match of a query iris sample.

2.4 Summary

This chapter addresses the problem of iris based human identification process in large databases. A specialized convolutional neural network architecture has been proposed which is trained as a Siamese architecture to learn compact feature vectors for iris images such that they have low inter-class and high intra-class similarity in the latent representation space. The extracted feature vectors are clustered using two techniques, k-means and agglomerative clustering to generate an index table. During retrieval, the candidates lying in the most similar index as that of the query image are fetched out for comparison. The generated candidate set is quite small in size as compared to the original database. The proposed technique has been tested on CASIA Interval and CASIA Lamp datasets and has been found to achieve 99% hit rate at just 2.254% and 0.008% penetration rate respectively. In other words, one needs to search only 2.254% and 0.008% of the considered datasets to be 99% sure about the presence of a sample in the database. A speedup of 4 and 27 times has been achieved when the CASIA Interval and CASIA Lamp have been indexed using the proposed method as compared to the naive approach for identification. The next chapter discusses a novel deep learning based indexing technique for palmprint databases.

Chapter 3

PalmHashNet: Palmprint Database Hashing Network

Palmprint is an impression procured from the inner part of the hand extending between the wrist and ends of fingers. Recently, palmprint-based biometric authentication gained wide popularity due to its non-intrusive nature, easy acquisition, and robust textural features [81]. A palmprint consists of complex and unique patterns which are utilized for human authentication. The features are classified as high resolution and low resolution based on the quality of the acquired image. High-resolution image refers to having 400 dpi or more, while low resolution refers to having 150 dpi or less [5]. The low-resolution features include wrinkles, texture, and principal lines. There are three principal lines in the palmprint $viz.$ heart, head, and life line. The low-resolution features are visible to the naked eye as well. On the other hand, ridges, singular point, and minutiae points can only be extracted from high-resolution images [22]. The high-resolution images are required in high-security areas or for criminal investigation. Low-resolution images are more suitable for civil and commercial applications for access control. These features are illustrated in Figure 3.1.

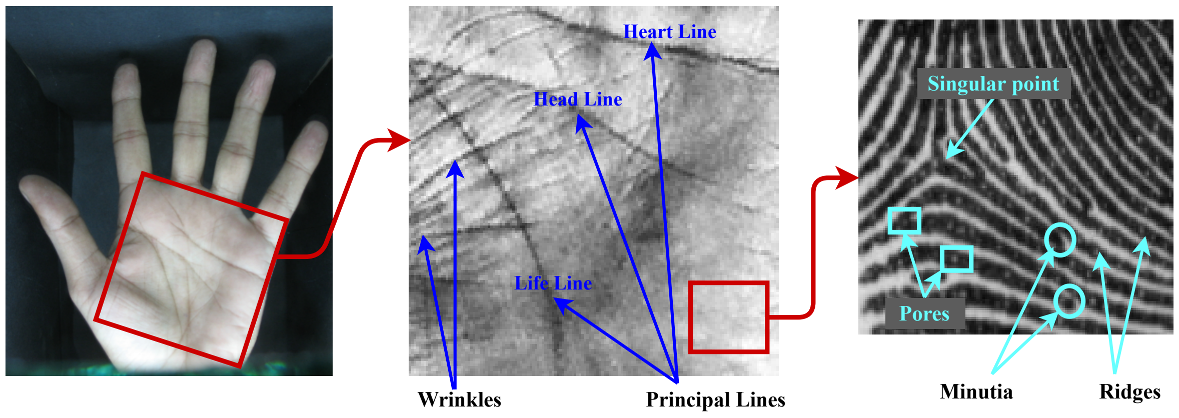


Figure 3.1: Palm image, ROI and features. The red box in first image represents the region of interest (RoI) of palmprint extracted from the hand image. The second and third image shows features on low and high resolution image.

Palmprint identification system [82] determines the identity of a given palmprint by comparing it with all the templates stored in a database. To reduce the number of comparisons for identification, the palmprint database is indexed such that feature vector of each palmprint image is associated with an index in the index table. The index table is generated such that the similar feature vector should lie in the bucket and the different feature vectors should be in separate buckets. Therefore, the learned feature vectors are expected to have high intra-class and low inter-class similarity. Hence, the performance of any biometric indexing technique is determined by the quality of the extracted features. The feature extraction process is carried out by training the network for a classification problem using softmax cross entropy loss [42]. The features are extracted from the layer connected just before the fully connected layer. It is believed that the learned feature vectors are good if they are able to classify the input image correctly. But, the learned features may not turn out to be optimal if no explicit constraint is applied on feature vector distribution which may lead to a general spread of the learned feature vectors. To address this, metric-based methods have been introduced that uses distance-based criterion to separate feature embeddings from different classes and bring them closer otherwise.

This chapter proposes a novel metric-based palmprint feature extraction network that uses ‘additive margin loss’ [83] to supervise the training process. The softmax loss is capable of maximizing the inter-class distance among samples of different classes but unable to minimize the intra-class dissimilarity among the samples of the same class. Therefore, a margin is added to the loss function to handle the intra-class variation better. It ascertains that the index space distribution is regularized to be similar to the uniform distribution. The network learns a 512-dimensional distinct compact feature embedding corresponding to every palmprint sample and is associated with an index in the index table. Two different techniques $viz.$ k-means clustering and locality sensitive hashing (LSH) with k-nearest neighbor search have been explored for index table creation. The generated index is used for the retrieval of top-k matches for identification. The indexing performance of both the considered techniques has been compared. The proposed technique is evaluated on four publicly available databases $viz.$, CASIA [49], IIT Delhi Touchless [50], Tongji Contactless [51] and PolyU II [52] palmprint dataset. All the databases contain palmprint images collected in an unconstrained environment. The results show the efficiency of the learned features in the identification process.

Next section in this chapter overviews state-of-the-art techniques that are proposed in the literature for palmprint recognition and palmprint database indexing. It is followed by the proposed technique, which further consists of three subsections $i.e.$, feature extraction, indexing, and retrieval. The last section gives details about the experimental setting to evaluate the proposed palmprint database indexing technique. It includes databases specifications, training and testing protocol, evaluation parameters and system specification. Lastly, the section discusses the obtained results for recognition and identification when indexing is applied on the considered palmprint databases.

3.1 Literature Survey

This section surveys various techniques for palmprint recognition and palmprint database indexing. Recognition performance of the system has also been evaluated to assess the quality of the learned features.

3.1.1 Palmprint Recognition

Palmprint recognition started with Boles et al. [84], wherein the authors proposed that palm shape and palm lines can be used for human authentication. Zhang et al. [85] claimed that using principal lines only for matching palmprint samples is not a good idea as some people may have similar patterns of principal lines. Therefore, they utilized a circular Gabor filter to extract features from low-resolution palmprint images. Kong et al. [86] proposed a palmprint verification system based on orientation information in palmprint lines. The paper presented a competitive code by extracting orientation information using 2-D Gabor filters. Angular matching was used to compare the generated codes. An improvised version of competitive code called robust line orientation code (RLOC) was proposed by Jia et al. [87]. Feature extraction was performed using a modified version of radon transform. The features were matched using pixel-to-area comparison. Zuo et al. [88] extended competitive code and proposed multi-scale orientation palmprint feature extraction called sparse multi-scale competitive code (SMCC). The proposed method is robust to illumination and scale variation. A palmprint verification method combining dominant orientation code and side code (DRCC) has been proposed in [89]. The proposed technique extracted both the codes by applying weights on the Gabor filter responses to improve the results. A local binary pattern-based feature descriptor (LLDP) to extract local features from palmprint images has been proposed in [90]. Li et al. [91] proposed a local feature descriptor that takes into account the direction and thickness information, making it robust to translation and rotation. Zhong et al. [92] proposed a siamese network utilizing two weights sharing VGG-16 networks to learn discriminative features for palmprint images. A histogram feature descriptor has been proposed in [93]. The paper suggested using apparent direction and latent direction computed from the energy map of the apparent direction. These directions are combined to form a single feature descriptor for palmprint images. Zhao et al. [94] proposed a CNN-based local feature extraction network in which a palmprint image is divided into five parts, and these parts along with the complete palmprint image is given to the proposed network. Zhong et al. [95] proposed a palmprint recognition system by combining large margin cosine loss and center loss.

3.1.2 Palmprint Indexing

The first technique for palmprint identification was proposed by You et al. [96], where a hierarchy of four-level features were used. The paper used different matching strategies at different stages while reducing the search space. Li et al. [97] proposed a palmprint retrieval technique by using texture features. The searching was performed in two stages; firstly, global features were used to find a small-sized candidate list, and then local features were used to output the final result from the selected candidates. Paliwal et al. [98] made use of the Vector Approximation (VA+) file database to generate score based indexing scheme. A ridge features-based indexing technique was proposed by Yang et al. [99]. The paper first aligns all the palmprint images in a unified coordinate system and then uses ridge density and orientation information for indexing. Chen et al. [100] proposed a technique that outputs a binary feature vector and applied spectral hashing technique to index the feature embeddings. Yue et al. [101] proposed two techniques that used different features for hash-table creation. The first one is based on orientation pattern (OP), which refers to orientation features. While the other one used principal orientation patterns (POP) $i.e.$, orientation patterns that lie in the region of principal lines. An accelerated and improvised indexing technique that uses features generated from POP was proposed in [102]. A convolutional neural network (CNN) based feature extractor was proposed in [103]. The proposed network outputs a 128-d feature vector, and later, implemented supervised hashing. A method using the difference of block means has been proposed by Almagtuf et al. [104]. In this method, no feature extraction was performed. Rather, simple operations such as adding and subtracting overlapping blocks are used to compute palmprint code in each direction. Chen et al. [105] proposed a double-orientation feature to account for unstable orientation fields. It further used a window-based feature measurement for faster retrieval. A distillation-based loss function has been proposed in [106] that generates binary feature vectors for palmprint images.

Zhu et al. [107] proposed an adversarial metric learning technique to make the palmprint embeddings uniformly distributed over a hypersphere. This is done by utilizing distance metrics and confusion terms. The paper also introduced a new palmprint database that was collected in an unconstrained environment. Zhao et al. [108] proposed a deep convolutional neural network based technique that extracted features from the palmprint image and its patches separately. All the features were then combined to form a compact feature. Jia et al. [52] evaluated the performance of various hashing techniques for retrieval of palmprint images. The paper considered four supervised, unsupervised, and deep hashing methods each and PolyU II, PolyU M_B, HFUT, TJU, and PolyU 3D databases for evaluation. It was reported that column sampling based discrete supervised hashing (COSDISH) [109] performs best among other considered hashing techniques. Recently, an end-to-end CNN-based network that learns binary hash values for palmprint images has been proposed in [110]. It used structural and pixel-level features by adding a similarity measurement module after the last fully connected layer.

3.2 Proposed Technique

This chapter proposes a novel technique to index a palmprint database to accelerate the identification process. There are three stages in the proposed technique $viz.$ feature extraction, indexing of the extracted features, and lastly, retrieval of the suitable candidate list for comparison with the probe sample. A metric-based learning deep learning network, called PalmHashNet, is proposed for feature extraction. It learns discriminative feature vectors for given palmprint images. An index table is generated by implementing k-means clustering and Locality Sensitive Hashing (LSH) on the learned feature vector space. When a query palmprint is shown to an identification system that uses an indexed database, its feature vector is first extracted using the trained PalmHashNet. The query feature vector is compared with all the indices $i.e.$, cluster centers, or hash values of the index table to find the most similar bucket. All the candidates lying in that bucket are retrieved for comparison with the query feature vector. Therefore, the query palmprint image is compared with the retrieved candidate list, which is smaller than the complete

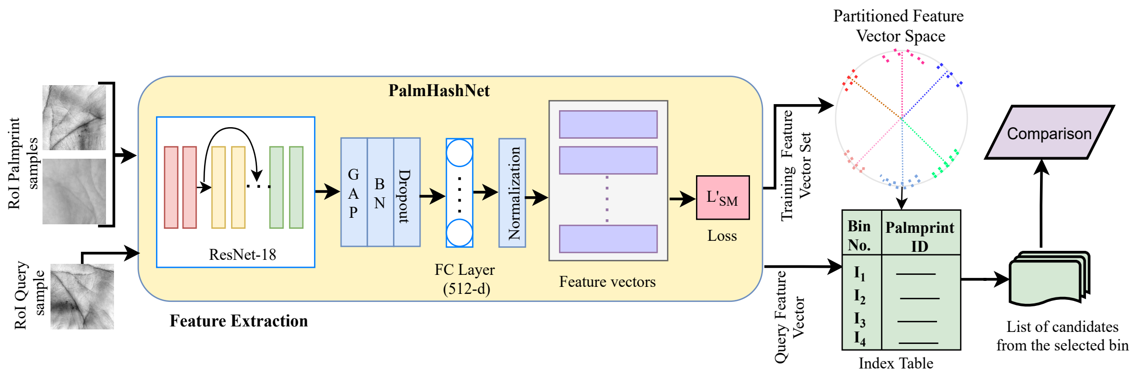


Figure 3.2: Block diagram of the proposed approach. There are three phases; PalmHashNet is the proposed feature extraction network that is trained using $L_{SM}^{'}$. The learned features are given to indexing module for index table creation. The feature of query palmprint sample is extracted using the trained PalmHashNet which is then compared with all the bin numbers and the most similar one is selected for candidate set generation.

database. The following subsections discuss the stages of the proposed technique.

3.2.1 Feature Extraction

The objective of feature extraction is to learn salient yet discriminative features that best represent a palmprint image. The performance of any biometric indexing technique is determined by the discriminatory ability of the extracted features. The features are expected to have a low inter-class and high intra-class similarity. The features belonging to the images of the same class should be closer to each other than the features belonging to the images of different classes in the feature embedding space. This condition is essential to improve search accuracy using the nearest neighbor approach. This results in making indexing and, further, the identification process efficient.

Generally, the feature extraction process is carried out by training a classification network using softmax loss [42]. Softmax loss is defined as an amalgamation of softmax function, cross-entropy loss, and the last layer of a convolutional neural network [111]. Different subjects or individuals are treated as different classes and the layer connected just before the fully connected layer serve as the feature embedding layer for the input image samples. Mathematically, softmax loss L_{SM} is defined as,

$$L_{SM} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{a_{y_i}}}{\sum_{j=1}^{C} e^{a_j}} \tag{3.1}$$

where N and C are the total number of samples and number of classes respectively. Activation of j^{th} neuron in the last fully connected layer having weight W_j and bias b_j for the i^{th} palmprint sample with feature f_i is given as $a_j = W_j^T * f_i + b_j$. There are C number of activations, one corresponding to each class. Let the ground truth for the i^{th} palmprint sample be the class y_i where $i \in \{1, 2, ..., C\}$, then the activation of the corresponding neuron can be given as $a_{y_i} = W_{y_i}^T * f_i + b_{y_i}$. Using this, the Eq.(3.1) can be written as,

$$L_{SM} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{W_{y_i}^T * f_i + b_{y_i}}}{\sum_{j=1}^{C} e^{W_j^T * f_i + b_j}} \tag{3.2}$$

Considering a binary classifier, the posterior probabilities of a palmprint having the feature vector f_i belonging to the class C_1 or C_2 can be obtained by using the softmax as shown in Eq.(3.3) and Eq.(3.4) respectively.

$$p(C_1) = \frac{e^{W_1^T * f_i + b_1}}{e^{W_1^T * f_i + b_1} + e^{W_2^T * f_i + b_2}} \tag{3.3}$$

$$p(C_2) = \frac{e^{W_2^T * f_i + b_2}}{e^{W_1^T * f_i + b_1} + e^{W_2^T * f_i + b_2}} \tag{3.4}$$

where (W_1^T, b_1) and (W_2^T, b_2) are the weight and bias corresponding to the class C_1 and C_2. The classifier outputs C_1 as the class of the query palmprint if $p(C_1) > p(C_2)$ and the output is C_2, otherwise. The classification solely depends upon the weight and bias term and uses $W_j^T * f_i + b_j$ for deciding the class. Element wise multiplication in $W_j^T * f_i$ is equivalent to the dot product therefore, the activation can be re-written as $a_j = ||W_j^T||\ ||f_i||\ \cos\theta_j + b_j$, where θ_j is the angle between vectors W_j and f_i. It can be observed that the activation depends on both the angle θ_j and the weight vector norm W_j. If the weights are normalized to unity, the classification becomes directly dependent on the angle between f_i and W_j's. Therefore, weight vectors and feature vectors are normalized as shown in Eq 3.5. Here, f_i and W_j are original feature and weight vector respectively. The values of $||f_i||$ and $||W_j||$ are set to unity using Eq 3.5.

$$f_i = \frac{f_i^\star}{\|f_i^\star\|} \quad \text{and} \quad W_j = \frac{W_j^\star}{\|W_j^\star\|} \tag{3.5}$$

After normalization, the posterior probabilities given in Eq 3.3 and Eq 3.4 can be equivalently changed to $p(C_j) = \cos\theta_j$ where $j = \{1, 2\}$. Therefore, the decision boundary becomes $\cos\theta_1 - \cos\theta_2 = 0$. Normalized features can be plotted on a hypersphere manifold with fixed radius, say 'r' as in shown in Figure 3.3. Softmax loss in the Eq.(3.2) can now be represented as,

$$L_{SM} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{r\,\cos\theta_{y_i}}}{e^{r\,\cos\theta_{y_i}} + \sum_{j=1, j\neq y_i}^{C} e^{r\,\cos\theta_j}} \tag{3.6}$$

While training, if the sample belongs to class C_1, the angle between f_i and W_1 is smaller than the angle between f_i and W_2. Although this kind of decision boundary works well for classification, it does not enforce high intra-class similarity as the compact localization of the same class features is not mandatory. Therefore, features obtained from different samples of the same class covering usual variation are scattered around feature space. This is a more visible phenomenon if samples contain high intra-class variations because of the presence of occlusion, pose, illumination *etc.*

In order to make the decision boundary more stringent, a margin m is added to θ. Consider a sample belonging to class C_1. This would imply that $\cos\theta_1 > \cos\theta_2$. By adding a margin m in θ_1, the equation changes to $\cos(\theta_1 - m) > \cos\theta_2$. The expression $\cos(\theta_1 - m)$ is larger than $\cos\theta_1$ which in turn is greater than $cos\theta_2$. The same relationship exists between θ_1 and θ_2. The decision boundary for class C_1 becomes $cos(\theta_1 - m) = cos(\theta_2)$. Similarly, to correctly classify the another feature belonging to class C_2, it is required that $cos(\theta_2 - m) > cos(\theta_1)$ and the decision boundary becomes $cos(\theta_2 - m) = cos(\theta_1)$. Due to the margin, lower bound of $cos\theta_1$ becomes much greater than $cos\theta_2$ thereby, enforcing higher intra-class compactness. The modified softmax function is written as below.

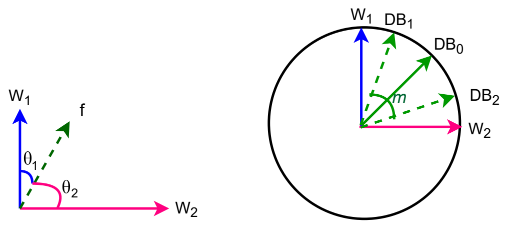


Figure 3.3: Geometrical Representation of (a) softmax loss and (b) additive margin loss. DB_0 is the decision boundary created by the softmax loss whereas DB_1 and DB_2 are the decision boundaries learned by additive margin loss for class C_1 and C_2 respectively. The boundary becomes a regional margin instead of single vector when additive margin is applied.

$$L^{'}_{SM} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{r\,(\cos\theta_{y_i}\,-m)}}{e^{r\,(\cos\theta_{y_i}\,-m)}+\sum_{j=1\,,\,j\neq y_i}^{C}\;e^{r(\cos\theta_j)}} \tag{3.7}$$

The geometrical representation of additive margin loss is shown in Figure 3.3. It shows that the initial decision boundary $i.e.$ the one created by softmax is now changed to DB_1 and DB_2 respectively for class C_1 and C_2 respectively. Therefore, we can conclude from the figure that intra-class difference among the samples of the same class is minimized by adding this extra marginal region to the angle.

3.2.2 PalmHashNet

This chapter utilizes a deep convolutional network named PalmHashNet as a feature extractor for palmprint images. The network is trained using the modified softmax loss function, given in Eq.(3.7). It has been observed that deeper networks result in loss of information resulting in stagnation of accuracy. To understand this problem, let us consider two networks, shallow and deep. The deep network is a superset of the shallow one $i.e.$, it consists of a shallow network with some additional layers that act as an identity function. This ensures that the deeper network acts just like its shallower counterpart in the worst-case scenario. However, it may happen that the deep network would learn better features and reduce the error significantly. Such networks are called residual networks, which are widely popular for image classification, and they consist of a series of residual units. Residual connections are added in the network to retain information of the previous layers to eliminate the problem of vanishing gradients in deeper networks. The proposed technique uses ResNet-18 as the backbone architecture to learn important yet discriminative features from palmprint images for indexing. The feature extraction process is carried out by training the ResNet-18 with the softmax loss. There are 16 convolution layers, two max-pooling layers, and a fully connected layer in ResNet-18 architecture. The filter size of the first convolution layer is 7×7 while in other layers, it is 3×3. This is followed by a global average pooling layer and a batch normalization layer. The global average pooling (GAP) layer aggregates the input features by taking an average across the channels. This consolidation brings down the requirement of the number of parameters and thus, reduces the chances of over-fitting. The output feature becomes robust to spatial translations of the input images [112]. Activations of the GAP layer output are fed to the batch-normalization (BN) layer [113] which normalizes the input by subtracting it by mini-batch mean and diving by the mini-batch standard deviation. Mini-batch refers to a subset of the training data that is given to the network in one epoch. Batch normalization smoothens the landscape of the loss function by bringing the spread of all the input dimensions to the neurons to the same distribution, resulting in faster training of the model. A dropout layer of 512 neurons has been introduced, followed by a fully connected layer to avoid over-fitting in the network. The last fully-connected layer serves as the feature embedding. The weights of this layer, along with the feature embeddings, are normalized. This makes the classification process solely dependent on the angle θ formed between the feature vector and weight vector. PalmHashNet is then trained in an end-to-end manner using the modified softmax loss, mentioned in Eq.(3.7). The architecture is shown in Table 3.1. PalmHashNet learns feature embeddings that have more intra-class and less inter-class similarity. The learned feature vectors are fed to the indexing module for index-table creation explained in the next sub-section.

Table 3.1: Architecture of the proposed feature extraction network (PalmHashNet) with respect to PolyU II database images as input.

Layer	# Filters	Output Shape	# of Param
Input	-	$128 \times 128 \times 3$	0
Conv2D - BN - PRelu	64	$128 \times 128 \times 64$	705
Maxpool 2D - BN	-	$64 \times 64 \times 64$	128
(Conv2D - BN - PRelu)*2	64	$64 \times 64 \times 64$	73986
Residual Block -BN	-	$64 \times 64 \times 64$	128
(Conv2D - BN - PRelu)*2	64	$64 \times 64 \times 64$	73986
Residual Block -BN	-	$64 \times 64 \times 64$	128
Conv2D - BN - PRelu	64	$64 \times 64 \times 64$	36993
Conv2D - BN	128	$32 \times 32 \times 128$	73984
Conv2D - BN - PRelu	128	$32 \times 32 \times 128$	8349
Residual Block -BN	-	$32 \times 32 \times 128$	256
(Conv2D - BN - PRelu)*2	128	$32 \times 32 \times 128$	295426
Residual Block -BN	-	$32 \times 32 \times 128$	256
Conv2D - BN - PRelu	128	$32 \times 32 \times 128$	147713
Conv2D - BN	256	$16 \times 16 \times 256$	295424
Conv2D - BN - PRelu	256	$16 \times 16 \times 256$	33281
Residual Block -BN	-	$16 \times 16 \times 256$	512
(Conv2D - BN - PRelu)*2	256	$16 \times 16 \times 256$	1180674
Residual Block -BN	-	$16 \times 16 \times 256$	512
Conv2D - BN - PRelu	256	$16 \times 16 \times 256$	590337
Conv2D - BN	512	$8 \times 8 \times 512$	1180672
Conv2D - BN - PRelu	512	$8 \times 8 \times 512$	132097
Residual Block -BN	-	$8 \times 8 \times 512$	1024
(Conv2D - BN - PRelu)*2	512	$8 \times 8 \times 512$	4720642
Residual Block -BN	-	$8 \times 8 \times 512$	1024
Dropout	-	$8 \times 8 \times 512$	0
MaxPool 2D	-	$4 \times 4 \times 512$	0
GAP - BN	-	512×1	0
Dropout - BN	-	512×1	0
Fully Connected	-	512×1	0

3.2.3 Indexing

Identification aims at finding the closest or most similar sample from the database for a query palmprint sample. The naive approach for identification involves comparing the query sample with all the images in the database and sorting the similarity score list to find the most suitable match. However, this process becomes computationally expensive with an increase in the size of the database. Therefore, there is a need to reduce the number of comparisons by reducing the search space for efficient identification. The database is indexed by associating the extracted feature vectors to an index. The query sample is compared with all the indices of the index table and the candidates belonging to the most similar index are retrieved for comparison. This proposed technique uses k-means clustering [114] and Locality Sensitive Hashing [115] for indexing the considered palmprint databases. The algorithm for feature extraction and indexing is given in Algorithm 3.

1. **k-means Clustering:** The objective of k-means clustering algorithm is to partition a set of feature vectors into a specified number of disjoint groups. Let there be a feature vector set $F = \{f_{P_1}, f_{P_2}, \ldots, f_{P_N}\}$ where, $P_1, P_2, \ldots, P_N$ represent N palmprint samples. k-means results in splitting F into 'k' disjoint clusters $c_1, c_2, ..., c_k$ such that similar feature vectors lie in the same cluster while separating those that are different from each other in the feature vector space. Each cluster has a representative data point that is also known as mean of the feature vectors lying in that particular cluster. The algorithm starts by initializing 'k' centers using k-means++ initialization [75]. Let the centers of 'k' clusters are represented by $m_1, m_2, \ldots, m_k$. k-means is a distance-based clustering algorithm which computes euclidean distance between a feature vector f_{P_i} where, $i = \{1, N\}$, and every cluster center. This helps in determining the closest cluster and f_{P_i} is assigned to that cluster. The cluster centers get updated after every iteration by computing the mean of all the feature vectors assigned to it. The process of assigning feature vectors to a cluster center and updating the cluster centers after each iteration is repeated till no further change in cluster assignment is observed or maximum number of itera-

tions have been exhausted. The goodness of clustering, that refers to how well k clusters approximate the feature vectors set, is evaluated by computing intra-cluster variance. Intra-cluster variance measures the amount of spread observed among the feature vectors lying in a particular cluster. It is computed using,

$$E_{k-means} = \sum_{j=1}^{k} \sum_{i=1}^{N} ||f_{P_i} - m_j|| \tag{3.8}$$

where, j and i denote number of clusters and feature vectors respectively. m_j is the center of cluster 'j'. After convergence, the cluster centers are stored as hash values in the index table and the palmprint IDs along with their feature vectors that lie in a particular cluster are stored in the bucket corresponding to it.

2. **Locality Sensitive Hashing** A Locality Sensitive Hashing (LSH) function maps the feature vectors to a lower-dimensional representation such that the feature vectors that are similar to each other are mapped in the same bucket with a high probability in the lower-dimensional space. The main objective of LSH is to maximize the probability of collision of similar items $i.e$, the probability of two similar feature vectors lying in the same bucket should be high. The hash function for an input feature vector f_{P_i} is computed by using two random values, $\vec{r}$ and x. Here, r is a d-dimensional vector whose entries are randomly chosen from a set of vectors following the Gaussian distribution. The dot product is quantized into a set of hash bins with the objective that all the nearby feature vectors should lie in the same bucket as shown,

$$h^{r,x}(f_{P_i}) = \left\lfloor \frac{\vec{r} \cdot f_{P_i} + x}{w} \right\rfloor \tag{3.9}$$

In this equation, w is the quantization width and x is a random variable lying between 0 and w. Quantization width determines the number of entries or candidates that would lie in each bucket of the hash table. Increasing the quantization width results in compact table as each bucket will have more number of entries. On the other hand, lower value of w results in larger table with lesser number of candidates in each bucket of the hash table. The search for true match for the query image is accomplished in a linear manner $i.e.$, the query image is compared with all the candidates lying in the selected bucket of the index table. Therefore, there is a trade-off between the table size and final number of comparisons.

Two conditions must be satisfied to serve the purpose of reducing number of comparisons for identification of a query palmprint sample. These are,

- The probability of two feature vectors lying in the same bucket of index table should be high if they are close to each other in the feature embedding space. Let there are two feature vectors represented by f_{P_1} and f_{P_2}. Let the euclidean distance between the two feature vectors is $< d_1$. This distance is $\leq d_1$ which is the threshold distance value that determines if the given two feature vectors are close to each other in the feature embedding space. In this case, both f_{P_1} and f_{P_2} will lie in the same bucket. This is mathematically represented as,

$$P[h(f_{P_1}) = h(f_{P_2})] \geq p_1 \qquad \text{if } \|f_{P_1}, f_{P_2}\| = d_1 \leq d \tag{3.10}$$

- Contrary to the previous condition, this condition states that the probability of two dis-similar feature vectors, f_{P_1} and f_{P_3}, lying in the same bucket should be low. Let d_1 and d_2 be the euclidean distance between f_{P_1} and f_{P_2} and f_{P_1} and f_{P_3} respectively. Since f_{P_1} and f_{P_3} are dis-similar feature vectors, the distance between them should be greater than d $i.e.$, $d_2 \geq a \times d$, where a is any constant. Therefore, the probability of them lying in the same bucket of the index table will be less. Mathematically, it can be shown as,

$$P[h(f_{P_1}) = h(f_{P_3})] \leq p_1 \qquad for \; \|f_{P_1}, f_{P_3}\| = d_2 \geq a \times d \tag{3.11}$$

To further increase or reduce the probability given in the conditions respectively, a

Algorithm 3 Feature Extraction and Indexing

```
Input: Set of palmprint images ($P = P_1, P_2, ..., P_N$)
Output: Index table $I$
1: for each palmprint sample $P_i$ do
2:     Extract feature $f_{P_i}$ using the trained PalmHashNet.
3:     Append $f_{P_i}$ to feature set $f_P$ $i.e.$ $f_P \leftarrow f_{P_i}$
4: Apply k-means clustering or Locality Sensitive Hashing on the feature vector set $f_P$.
5: for each cluster center $cc_i$ (k-means) or hash value $h_i$ (LSH) do
6:     Create an entry in index table $I$ with $cc_i$ or $h_i$.
7:     Put ID and feature vectors of all the candidates in $I$ lying in $cc_i$ or $h_i$.
return I
```

hash function of t- bits can be generated by performing t dot products in parallel using Eq.(4.16). A $t-$bit hash value of a feature vector belonging to palmprint sample P_1 can be computed by concatenating t values determined using the Eq.(4.16). $h(f^t_{P_1})$ can be written as $= h_1(f_{P_1}), h_2(f_{P_1}), \dots, h_t(f_{P_1})$. After implementing LSH on the set of feature vectors generated by the trained model, we get a data structure that consists of hash value and the candidate IDs in its corresponding bucket.

3.2.4 Retrieval

The objective of the retrieval stage is to return a list of candidates that could be probable matches of a query palmprint image. This starts off by extracting feature vector of the query image q_i using the proposed PalmHashNet. In case of index table generated through k-means, the centers of each cluster act as indices. Therefore, the query feature vector is compared with all the cluster centers or indices of the index table. The one which has the highest similarity is selected and candidates contained in that cluster are retrieved for identification. On the other hand, in case of locality sensitive hashing, a $t-$bit hash function corresponding to the query feature vector is computed using Eq.(4.16). The hash bucket which is the most similar with the computed hash function is selected from the generated index table. The process is explained in Algorithm 4. This is followed by identification that aims at finding true match of the query sample from the retrieved candidate set. All the retrieved candidate IDs' feature vectors are matched with the query feature vector and their similarity scores are obtained. The rank of true match is determined from

Algorithm 4 Retrieval

```
Input: Index table $I$
Output: Candidate Set $X$
1: for each query palmprint sample $q_j$ do
2:     Extract feature $f_{q_j}$ using the trained PalmHashNet.
3:     for each index $i$ in $I$ do
4:         Compute cosine similarity between hash value ($h_i$) and $f_{q_j}$.
5:         Create a table $S$ with $h_i$ and cosine similarity.
6:     Find the maximum score value.
7:     Retrieve IDs stored in the most similar bin to get candidate list $X$ for matching.
return X
```

the sorted score list file to evaluate the performance of the proposed indexing approach. This process makes identification a constant time operation as the size of the retrieved candidate list is fixed.

3.3 Experimental Results

This section gives details about databases specifications, experimental setting and training and testing protocol considered for palmprint database indexing. The results are obtained for both palmprint recognition and palmprint identification with indexed database.

3.3.1 Database Specifications

Four publicly available standard databases namely, CASIA Palmprint Image Database, IIT Delhi Touchless Palmprint Database, Tongji Contactless Palmprint Dataset and Hong Kong Polytechnic University Palmprint II Database (PolyU II) have been used to evaluate the performance of the proposed technique. The details of the databases are given below. Region of interest (RoI) samples from each database is shown in Figure 3.4.

- **CASIA Palmprint Image Database [49]:** This database, also referred to as CASIA-Palmprint, consists of 5502 palmprint images collected from both left and right hand of 312 individuals. Eight images from each subject were collected in single session. The individuals were not instructed regarding positioning their hands in a specific way during acquisition. Therefore, the acquired images have huge pose variance. Palmprint RoI of size 128×128 pixels is segmented out from the acquired images for experimentation.

- **IIT Delhi Touchless Palmprint Database [50]:** IITD palmprint database was collected from students and staff of Indian Institute of Technology Delhi, India in June 2006 and July 2007. A total of 2600 images were collected from 460 palms of 230 subjects. All the images are in bitmap format and the RoI has a resolution of 150×150 pixels.
- **Tongji Contactless Palmprint Dataset [51]:** This dataset is comparatively larger in size than the aforementioned two databases. It consists of palmprint images collected from both hands of 300 individuals. Ten images of each palm per individual were acquired in two separate sessions, making it a dataset of total 12000 images. The extracted RoI has a resolution of 128×128 pixels.
- **Hong Kong Polytechnic University Palmprint II Database (PolyU II) [52]:** This database consists of 7752 palmprint samples that were collected from 193 individuals. The samples have been collected in two different sessions with a gap of 2 months. Ten palmprint samples from each palm of all the individuals were acquired. The extracted RoI has a size of 128×128 pixels.

3.3.2 Training and Testing Protocol

The images contained in the considered datasets are not uniform and there is no standard training and testing protocol that is associated with the datasets. Therefore, the proposed technique adopts the mostly followed training and testing partition for the considered databases. The training partition refers to the gallery images or the samples stored in database. On the other hand, testing partition contains the query images on which identification needs to be performed. The CASIA-palmprint database is partitioned into 80%$-$20% split for training and testing respectively. In IIT Delhi palmprint database, each subject has either given five or six images. Hence, we have used 4 images for training and the remaining 1 or 2 images are used as query samples for testing. For the Tongji

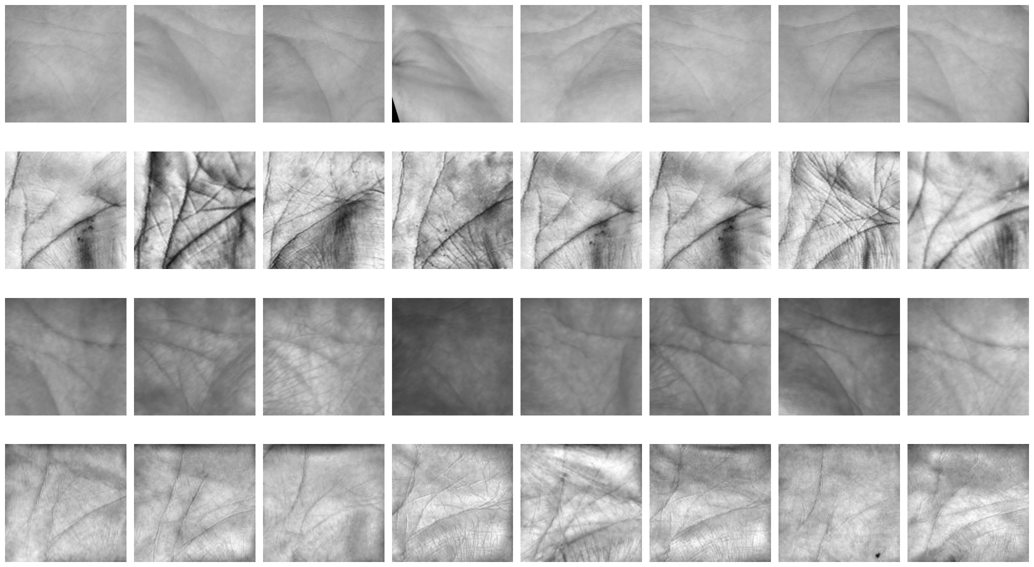


Figure 3.4: Sample palmprint region of interest (RoI) images from the CASIA-Palmprint, IIT Delhi Touchless, Tongji Contactless, and PolyU II Palmprint Dataset (row wise respectively).

contactless and PolyU II database, the training and testing split is done session-wise $i.e.$ 10 images from session 1 are used as gallery images while the remaining 10 images collected in session 2 are used as query images.

3.3.3 Experimental Setting

The proposed PalmHashNet has been implemented using Pytorch library of Python programming language. Lately, the palmprint acquisition is usually accomplished in an unconstrained environment for better user experience. The acquired images tend to have occlusion, illumination, pose variation etc. To make PalmHashNet robust to such variations, data augmentation is applied on the training partition of the databases. It helps in making the deep neural network more robust by training it on different variations of the palmprint samples. Python Augmentor library [116] has been used to augment the training partition by applying image transformation operations such as zoom, distortion, rotation and illumination. With these four operations applied on each image, four new images were created. Thereby, making training partition grow by five times. Hence, the size of the training partition became 12480, 9205, 19300 and 30000 for CASIA, IITD-

Table 3.2: Database specification with the number of comparisons during verification.

	CASIA [49]	IITD [50]	PolyU II [52]	Tongji [51]
No. of subjects	312	460	193	300
Gallery samples	2496	1841	3860	6000
Probe samples	3006	760	3860	6000
Genuine comparisons	24048	3041	38600	60000
Imposter comparisons	7478928	1396118	14861000	35921592
Total comparisons	7481376	1399159	14899600	35981592

Touchless, PolyU II and Tongji Contactless databases respectively. The test partition has not been augmented as the test needs to be performed on the original images only.

For training the network and evaluating the indexing and retrieval performance, a computer with Xenon (R) processor with 32 GB RAM and 12 GB on card RAM on NVIDIA Tesla K40C GPU has been used. For index table creation, k-means and LSH have been utilized for partitioning the set of learned feature vectors. k-means is used to cluster similar feature vectors based on similarity for the purpose of index table creation. The ideal number of clusters, denoted by k, for a particular dataset is determined by using a metric called silhouette coefficient [117]. Silhouette coefficient evaluates the goodness of a clustering technique when the data is split in k clusters. Mathematically, it is defined as,

$$\text{Silhouette Coefficient} = \frac{q - p}{\max(p, q)} \tag{3.12}$$

where, p and q refer to the average intra-cluster and inter-cluster distance respectively. Intra-class distance is the distance between each point within the same cluster. On the other hand, inter-class distance is computed between data points lying in different clusters. The value of silhouette coefficient lies between -1 to $+1$ with $+1$ indicating well separated clusters and -1 indicating the opposite. To determine the appropriate value of k, the silhouette coefficient for various values of k is computed and it was experimentally observed that the its value is highest at $k = 60$. This became the initial point for finding the suitable value k for the indexing approach. Further, k-means was implemented for all values in the range interval of $[5, 100]$ with a difference of 5. It was empirically

Table 3.3: Recognition performance of PalmHashNet on the considered databases. Higher accuracy and DI are better, however EER is better when lower.

	CASIA [50]	**IITD** [49]	**Tongji** [51]	**PolyU II** [52]
Accuracy	99.98%	99.62%	97.85%	99.83%
EER	0.031%	0.39%	0.53%	0.011%
DI	4.71	3.94	2.82	4.10

determined that the performance of indexing achieved best results when $k = 65$ for the considered datasets. For LSH, a hash code of 5 bits is generated for each palmprint sample. This value is empirically determined after various experiments.

3.3.4 Results

To validate the performance of the proposed technique, results are computed for both verification and identification system. Firstly, the quality of the learned features are computed because it is their discriminating ability that determines the performance of the indexing module. Therefore, the recognition results are listed followed by indexing performance based on two different techniques namely, k-means clustering and Locality Sensitive Hashing (LSH).

3.3.4.1 Recognition Performance

To evaluate the verification performance of the proposed approach, we have computed Accuracy, EER and DI of the system. To compute these parameters, each sample in the test partition is compared with all the samples in the training partition. The total number of comparisons which comprise of genuine and imposter comparisons is given in Table 3.2. In the table, gallery images refer to training split as these are the images that are indexed while probe samples refer to the testing split as these images would be sent to the identification system for querying. The proposed technique obtained $> 99\%$ accuracy for three databases namely CASIA, IITD Touchless and PolyU II database. However, 97.85% accuracy is achieved for Tongji Contactless database. The obtained EER is $< 1\%$ for all the considered datasets. The obtained values of these parameters for the considered databases is shown in Table 3.3. With low values of EER and high values of accuracy, it is clearly evident that the proposed technique performs quite well in case

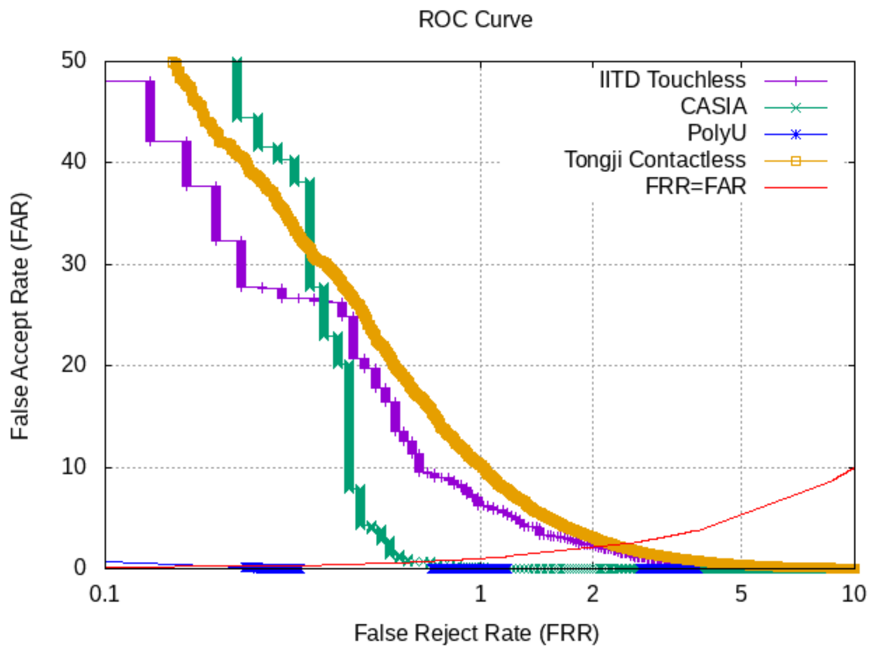


Figure 3.5: ROC of the proposed approach on the considered datasets.

of palmprint recognition. The Receiver Operating Characteristics curve of the proposed approach for all the considered datasets is shown in Figure 3.5. Here, $FAR = FRR$ line is not straight because FRR is plotted on log scale. The log scaling brings the focus towards more meaningful lower side of the curve. The area under the ROC curve depicts error and a system having lesser area is considered to be better in general. Different ROC curves shown in Figure 3.5 compare the classification performance of the proposed feature on the four databases $viz.$ CASIA, IITD-Touchless, Tongji Contactless and PolyU II databases. The recognition performance of the proposed technique is compared with various palmprint recognition techniques proposed in the literature. The same is shown in Table 3.4 and it is clearly evident that PalmHashNet performs best in terms of equal error rate when tested on CASIA, IITD Touchless and PolyU II database.

3.3.4.2 Indexing Performance

Indexing performance is evaluated in terms of hit rate and penetration rate. It is expected from a good indexing technique to achieve lower penetration rate at high hit rate. Table 3.5 and Table 3.6 show penetration rate required for different values of hit rate on

Table 3.4: Table showing Equal Error Rate (EER) comparison of the proposed approach (PalmHashNet) with state-of-the-art palmprint recognition techniques.

Author (s)	**Technique**	**CASIA**	**IITD**	**PolyUII**	**Tongji**
Lu et al. [85]	Competitive Code (CC)	0.55	7.72	0.6	-
Jia et al. [87]	Robust line orientation code	0.81	7.44	0.16	-
Zuo et al. [88]	Sparse multiscale CC	0.48	-	0.01	-
Luo et al. [90]	LBP based feature descriptor	-	4.07	0.02	-
Li et al. [91]	Direction and thickness	-	0.87	-	-
Zhong et al. [92]	Siamese network	-	-	0.28	-
Zhong et al. [95]	Large margin cosine loss	-	-	0.08	**0.25**
Zhu et al. [107]	Adversarial metric learning	-	1.73	0.86	1.21
Proposed	**PalmHashNet**	**0.03**	**0.39**	**0.01**	0.53

the considered four databases when k-means and LSH are implemented respectively for indexing. The proposed approach achieves a hit rate of 95% at 0.022% and 0.025% penetration rate for CASIA and PolyU II database irrespective of which technique has been applied for indexing. But for the other two databases, the penetration rate is manifold for k-means as compared to LSH. The penetration rate for k-means is increasing very fast while it remains low for LSH at even high hit rates. Therefore, it can be concluded that LSH performs better than k-means clustering for index table creation. Hence by considering LSH, we can conclude that 100%, 98.81%, 97.08% and 100% hit rate is achieved at rank-1 for CASIA-Palmprint, IITD Touchless, Tongji Contactless and PolyU II palmprint databases respectively. The results signify that to achieve true match in these databases with 100% confidence, it is required to compare the query image with only **0.800%, 1.032%, 4.555% and 0.39%** of the respective databases. Cumulative Match Curve (CMC) is a rank-based metric that shows relationship between identification probability at a given rank. That is, what ratio of total queries got correctly identified till a particular rank. The CMC showing relationship between probability of identification and rank for the proposed technique with both k-means and LSH is shown in Figure 4.9.

3.3.4.3 Time analysis

The time-based performance of the proposed approach is evaluated in terms of speed-up. It refers to how fast the identification process has become by using the proposed approach

Table 3.5: Penetration rates (%) for different values of Hit Rate (HR) for the proposed approach when k-means clustering is applied for indexing.

HR(%)	CASIA [50]	IITD [49]	Tongji [51]	PolyU II [87]
90	0.022	0.054	6.55	0.025
95	0.022	2.39	13.15	0.025
96	0.022	3.096	15.633	0.025
97	0.022	4.943	19.166	0.025
98	0.022	7.17	23.089	1.295
99	0.022	17.599	32.866	2.875
100	3.367	34.818	69.983	20.777

Table 3.6: Penetration rates (%) for different values of Hit Rate (HR) for the proposed approach when LSH is applied for indexing.

HR(%)	CASIA [50]	IITD [49]	Tongji [51]	PolyU II [87]
90	0.022	0.054	0.016	0.025
95	0.022	0.054	0.016	0.025
96	0.022	0.054	0.016	0.025
97	0.022	0.054	0.02	0.025
98	0.022	0.054	0.12	0.025
99	0.022	0.162	0.87	0.025
100	0.800	1.032	4.555	0.39

for indexing as compared to the naive approach for identification (without indexing). It is calculated by the total time taken to find a suitable candidate set from the created index table for matching and identifying a query image. The time is measured in seconds (s). The proposed approach takes 0.0013s, 0.018s, 0.040s and 0.010s for identification on CASIA, IITD Touchless, Tongji Contactless and PolyU II database respectively using the traditional approach. However, this time reduced to only 0.0008s, 0.006s, 0.020s and 0.0037s respectively when the proposed indexing technique is applied. Table 3.7 shows a comparison of the average query time taken for palmprint identification with and without indexing (using LSH) on the considered datasets. It can be observed that the average speedup is 6 times if the proposed approach is used for identification on the considered four databases.

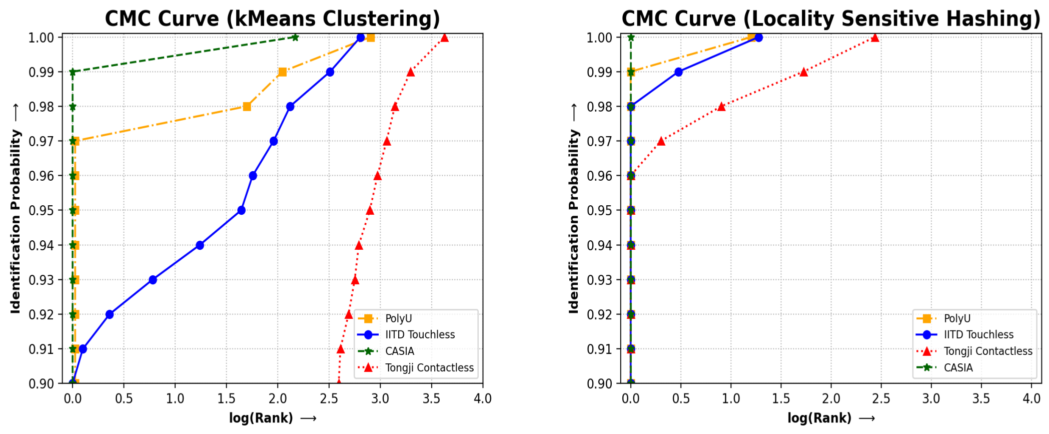


Figure 3.6: CMC curve showing relationship between Identification probability and rank in case of k-means clustering and locality sensitive hashing.

Table 3.7: Query time (in seconds) comparison of the proposed approach against non-indexed process for identification on the four databases.

Database	Without Indexing	With Indexing
CASIA [50]	0.013	**0.0008**
IITD [49]	0.018	**0.006**
Tongji [51]	0.040	**0.020**
PolyU II [87]	0.010	**0.0037**

3.3.4.4 Comparative analysis

Rank-1 identification rate has been used to compare the proposed technique with state-of-the-art techniques. One comparison is made with the available supervised, unsupervised and deep hashing techniques on Tongji Contactless and PolyU II databases [52]. The study [52] demonstrates the performance of different hashing techniques on palmprint databases. It was reported that rank-1 identification rate for KNNH [118], DSH [119] and ADSH [120] was 96.25%, 91.4% and 96.05% for Tongji Contactless database while the values were 98.30%, 93.35% and 94.86% for PolyU II palmprint database. However, COSDISH [52] achieved best results with 96.38% and 99.25% rank-1 identification rate on these two databases respectively. However, the proposed approach achieves rank-1 identification rate equal to 97.08% and 100% on Tongji contactless and PolyU II database, surpassing other hashing techniques. Another comparison is made with state-

Table 3.8: Table showing Rank-1 identification rate (I.R.) comparison of the proposed approach (PalmHashNet) with state-of-the-art techniques on various databases.

Author	**Feature**	**CASIA**	**IITD**	**Tongji**	**PolyU II**
Paliwal et al. [98]	Vector Approximation (VA+) file database	-	-	-	96.01 %
Almaghtuf et al. [104]	Difference of block means	94.17%	99.02%	-	99.40%
Zhu et al. [107]	Adversarial metric learning	-	99.02%	97.71%	99.02%
Zhao et al. [108]	Deep feature by combining local and global features	97.06%	97.25%	-	-
Jia et al. [52]	Column sampling based discrete supervised hashing	-	-	96.38%	99.25%
Chen et al. [105]	Double-orientation feature from orientation fields	-	-	-	99.25%
Liu et al. [110]	Similarity Metric Hashing Network	-	-	97.65%	-
Proposed	**PalmHashNet**	**100%**	**99.42%**	**97.08%**	**100%**

of-the-art palmprint indexing techniques in Table 3.8. The cell values in the table indicate Rank-1 identification rate achieved by various techniques on the considered databases. A blank entry indicate that the corresponding technique is not evaluated on the mentioned database. The best rank-1 identification rate for CASIA was achieved by Zhao et al. [51]. For IITD Touchless and Tongji Contactless databases, the approach proposed by Zhu et al. [107] was giving best results till now. However, approach proposed in [104] performed best on PolyU II database. It is clearly evident from the Table 3.8 that our approach achieves best value of rank-1 identification rate for CASIA, IITD Touchless and PolyU II databases. Therefore, by considering both the comparisons, the proposed approach outperforms other techniques proposed in literature.

3.3.4.5 Ablation Study

The proposed approach solely depends on the feature extraction process and thus, it was required to find the best features that could perform well for recognition process as well as for identification. Therefore, two different ablation studies have been done in this chapter. The objective of first study is to check the effect of adding margin to the softmax loss for training the feature extraction model. The other aims to analyze the effect of different dimensions (sizes) of feature vectors to find the most appropriate size that best represents a palmprint image. To accomplish the first study, features extraction model was trained using only the softmax loss and then margin was added to see if there was any improvement with respect to the quality of the extracted feature vectors that further affects indexing and identification. Rank-1 identification rate was computed for the considered databases on the extracted features using only the softmax loss. Table 4.6 shows the comparison with respect to the rank-1 identification rate obtained by the proposed approach and the model trained with the softmax loss. It was observed that the Rank-1 identification rate improved by 10.29%, 14.53%, 4.59% and 2.75% times on CASIA, IITD-Touchless, Tongji Contactless and PolyU II databases respectively when additive margin loss was introduced in the feature extraction network.

Different dimensions of feature vectors such as 128, 256 and 512 have been considered to find the suitable feature vector size that best represents a palmprint image taken from the considered databases. Indexing is performed on the databases with respect to all the aforementioned feature vector dimensions using Algorithm 3 evaluates the identification performance. It was empirically determined that the system achieves higher accuracy with 512-dimensional feature vector. The same is shown in Table 3.10. A relationship between hit rate and penetration rate was established for all the combinations. A set of candidates for a query palmprint image is retrieved for comparison from the indexed palmprint database. Hit rate determines the confidence by which a query image can find its true match in the retrieved set of candidates. Penetration rate refers to the ratio of the database required to be retrieved for finding the true match of a query image. The true identity of a query sample is expected to be established by retrieving only a small

Table 3.9: Table comparing Rank-1 identification rate of model trained using softmax loss and model trained using the proposed methodology.

	CASIA	**IITD**	**Tongji**	**PolyU II**
W/o additive margin	89.71%	84.89%	92.49%	97.25%
Proposed	100%	99.42%	97.08%	100%

percentage of the database (low penetration rate) with high confidence (high hit rate). An efficient biometric indexing approach is expected to achieve high hit rate at lower value of penetration rate. The same has been shown in Table 3.10. It is clearly evident that 512-d feature vector performs best on mostly all the considered datasets. Hence, we have considered 512-d feature vector in this study. The graphs showing hit rate vs. penetration rate for all the experiments are shown in Figure 3.7.

3.4 Summary

This chapter proposes a palmprint database indexing approach called PalmHashNet that generates highly discriminative embeddings to create a fixed-size candidate list for comparison to make identification a constant time operation. Softmax loss with additive margin has been used to train the model for palmprint database indexing and to learn the feature vector embeddings simultaneously. This loss function ensures that the learned feature embeddings have low inter-class along with high intra-class similarity. The learned embeddings have been indexed using k-means Clustering and Locality Sensitive Hashing technique to create an index table. Identification experiments are conducted on four publicly available popular palmprint databases $viz.$ CASIA, IITD-Touchless, Tongji-Contactless and PolyU II palmprint databases. The proposed approach achieved a penetration rate of 0.022%, 1.032%, 4.555% and 0.39% at 100% hit rate for the respective databases. Therefore, it can be concluded that to find the true match of a query sample with 100% confidence, it is required to look out for $< 1\%$ of the CASIA and PolyU II database and 1.03% and 4.55% of the IITD-Touchless and Tongji Contactless database respectively. The proposed approach outperforms other state-of-the-art recognition and indexing techniques proposed in the literature.

Table 3.10: Ablation study: The table showing **Penetration Rate (%)** at various Hit Rate (%) w.r.t. different feature vector sizes (128-d, 256-d and 512-d) for the considered datasets.

			Hit Rate (%)						
Databas	**Indexing Scheme**	**Feature**	**90%**	**95%**	**96%**	**97%**	**98%**	**99%**	**100%**
IITD [50]	k-means	**128**	0.054	1.684	2.227	2.390	3.313	12.710	39.109
		256	1.683	3.747	5.268	8.636	12.873	19.391	33.134
		512	0.054	2.390	3.096	4.943	7.170	17.599	34.818
	LSH	**128**	0.054	0.054	0.054	0.054	0.054	0.054	4.940
		256	0.054	0.054	0.054	0.054	0.054	0.054	4.562
		512	0.054	0.054	0.054	0.054	0.054	0.162	1.032
CASIA [49]	k-means	**128**	0.070	0.070	0.070	0.070	3.132	3.132	73.337
		256	0.070	0.070	0.070	0.070	0.070	0.070	13.254
		512	0.070	0.070	0.070	0.070	0.070	0.070	9.678
	LSH	**128**	0.054	0.054	0.054	0.054	0.380	1.980	6.930
		256	0.054	0.054	0.054	0.054	0.380	1.730	7.827
		512	0.054	0.054	0.054	0.054	0.054	0.162	1.032
Tongji [51]	k-means	**128**	7.083	15.233	18.550	24.066	31.266	43.683	99.666
		256	7.016	15.680	18.833	24.150	31.266	45.033	89.550
		512	6.550	13.150	15.633	19.166	23.083	32.866	69.983
	LSH	**128**	0.016	0.016	0.016	0.036	0.238	0.827	5.645
		256	0.016	0.016	0.016	0.020	0.140	0.970	6.435
		512	0.016	0.016	0.016	0.020	0.120	0.870	4.555
PolyU [52]	k-means	**128**	0.025	0.025	0.025	0.025	1.554	2.927	28.626
		256	0.025	0.025	0.025	0.025	1.813	2.772	97.642
		512	0.025	0.025	0.025	0.025	1.295	2.875	20.777
	LSH	**128**	0.025	0.025	0.025	0.025	0.025	0.025	0.783
		256	0.025	0.025	0.025	0.025	0.025	0.025	0.310
		512	0.025	0.025	0.025	0.025	0.025	0.025	0.390

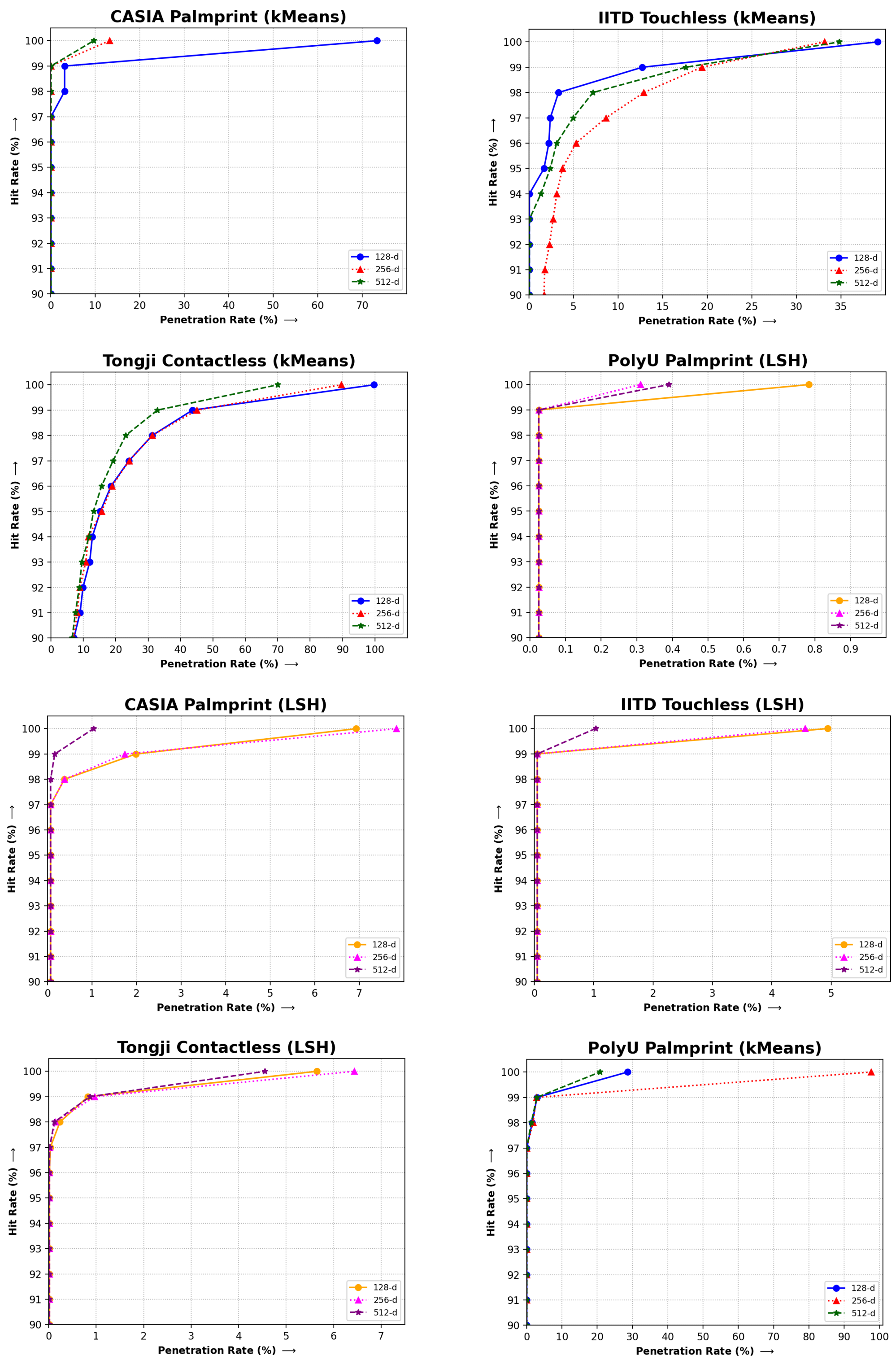


Figure 3.7: Ablation Study for different feature size on the considered datasets using k-means clustering and LSH for index table creation.

Chapter 4

FKPIndexNet: Indexing Finger-Knuckle-Print Database

Hand-based biometric traits such as fingerprint and palmprint are easy to acquire and are more widely used in authentication applications. However, in countries like India, where a significant amount of the population is involved in agriculture or labor activities, severe damage happens to the inner part of the hand. This results in deterioration in the quality of the acquired fingerprint or palmprint sample, leading to poor feature extraction and ultimately failure in the identification process [121, 122]. Finger-knuckleprint (FKP) is a better alternative for hand-based biometric authentication. FKP refers to the impression obtained from the outer surface around the phalangeal joint of a finger. It contains a rich texture that is unique among the population. FKP acquisition is easy as it needs less user cooperation and can be acquired from a distance with low-resolution cameras. Considering the present situation where the world is hit by a pandemic, a biometric trait acquired in a contact-less manner is more suitable. Therefore, finger-knuckleprint appears as a suitable alternate among hand-based biometric traits for authentication applications.

An FKP identification system aims at establishing the identity of a given FKP sample by comparing the input image with all the templates stored in the database. A similarity score is computed for every match, and the scores are sorted to find the true match of

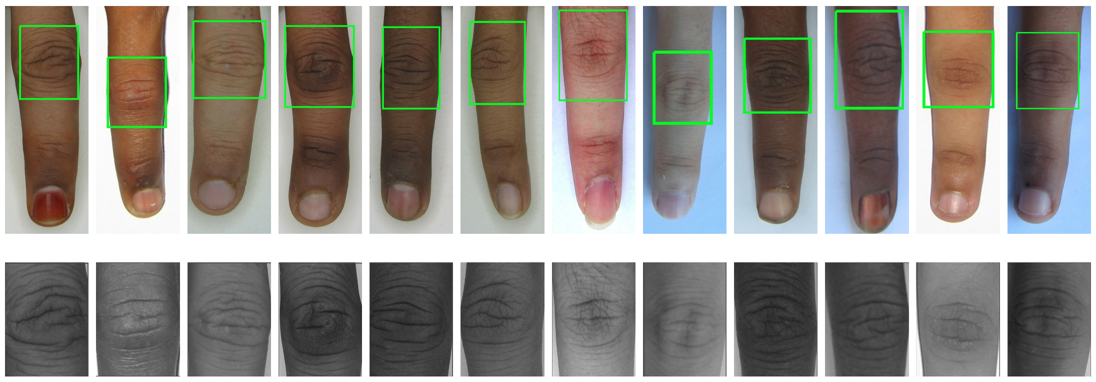


Figure 4.1: First row: Manually annotated finger-knuckleprint images from PolyU-FKP database. Second row: Output of FKPSegNet; Extracted region of interest gray-scale FKP images.

the query FKP sample. Hence, identification involves 1:N comparisons, where N is the number of templates stored in the database. The number of comparisons needs to be reduced by narrowing the search space to make the identification system faster and more efficient. This process of filtering out a subset of suitable candidates for comparison with the probe image to facilitate faster identification is known as Indexing. Indexing of a biometric database involves associating the feature vector corresponding to FKP samples with an index in the index table. A latent representation of an FKP image consists of its most salient and discriminating features. This step results in generating an index table consisting of FKP feature vectors and their subject IDs bind to an index.

This chapter proposes a novel approach for indexing the FKP database to facilitate faster and efficient identification. To our knowledge, this is the first work that aims to extract deep features from FKP images to be used for index table generation. Firstly, the dorsal finger images are given to the proposed FKPSegNet, which segments the region of interest (RoI) from it. Next, the feature extraction network is trained using the RoI images of the FKP database. A custom loss function is proposed that ensures that the learned latent representations have high intra-class and low inter-class similarity. It ascertains that the index space distribution is regularized to be similar to the uniform distribution. The proposed feature extraction network outputs a $512-$dimensional feature embedding corresponding to every FKP sample. The learned feature vectors are associated with an index, and an index table is generated. Three techniques $viz.$ k-means clustering, BallTree hashing, and locality sensitive hashing with the nearest neighbor search are explored for index table construction. The features are extracted for the query FKP sample using the trained feature extraction network. The extracted feature is compared with all the indices of the index table, and the candidates lying in the most similar one are retrieved for comparison. The proposed technique is evaluated on two publicly available databases $viz.$, PolyU-FKP [53] and IIT Delhi Finger Knuckle Database [54]. The proposed technique facilitates learning efficient latent representations that ensure fast and accurate retrieval during identification without compromising recognition accuracy. The rest of the chapter is organized as follows. The next section gives an overview of the related work that has been done in FKP recognition. Section 4.2 describes the proposed technique in detail. Experimental setting and results have been analyzed in Section 4.3.

4.1 Literature Survey

Automated human identification using finger-knuckleprint has gained a lot of research interest in the recent past. Woodward et al. [123] proposed for the first time that the surface of fingers can be used for biometric authentication. The authors also concluded that a finger surface is effective as a 2D face image for authentication. However, Kumar et al. [124] proposed that the back surface of the image consists of a unique texture and investigated its usage for authentication. The approach uses peg-free imaging and exploits the features from hand geometry to improve the recognition system's performance. Zhang et al. proposed an FKP recognition approach that uses Gabor filters to extract orientation features in [8]. The authors also set up an acquisition device for FKP data collection and collected 5760 images from 480 fingers. Another approach proposed by Zhang et al. [125] uses band-limited phase only correlation that eliminates high-frequency components which can be prone to noise. The authors also acquired a knuckleprint database consisting of 7,920 images from 660 fingers. An approach that uses local binary pattern (LBP) histograms for FKP recognition has been proposed in [126]. The approach divides the FKP image into several blocks and then devices Gabor filters over those blocks to generate LBP histograms. To address the large intra-class distance between the FKP samples, a reconstruction-based approach for FKP recognition has been proposed in [127]. The approach aims to learn a dictionary from the gallery FKP images, and each query sample is represented as a linear combination of the values in that dictionary. Later, the Competitive Coding technique is used to extract orientation features from the FKP images. Yu et al. proposed an LBP-based feature extraction technique for FKP matching in [128]. The image is divided into blocks, and LBP histograms are extracted and concatenated to represent the full FKP image. An integration technique that uses both orientation and texture features has been proposed in [129]. The paper addressed the possibility of multiple orientations by using multi-level thresholding that performs orientation coding corresponding to each Gabor filter response. The texture features are extracted using LBP. Lastly, both texture and orientation features are integrated using score-level fusion. Nigam et al. proposed curved Gabor filters for feature extraction from FKP images [23]. In the paper, two encoding schemes, namely, Gradient Ordinal Relation Pattern (GORP) and STAR GORP (SGORP), have been proposed to represent each FKP image. In GORP, each pixel is not represented by its gray value, and a code is obtained using the gradient of its eight neighboring pixels computed using x and y directions Scharr kernels. On the other hand, SGORP encodes the relationship between the top and bottom pixels along with the diagonally opposite ones. Usha et al. proposed an FKP recognition technique that uses shape-oriented features, as well as texture information [130]. The shape features are extracted using angular geometric analysis while the texture information is extracted using multi-resolution transform, also called Curvelet transform. In [131], a fast matrix projection method for extracting line features is used for verification. Support Vector Machine (SVM) has been employed with extracted long and short Gabor features to improve the recognition performance in [132]. To our knowledge, there has been only one work that aims at indexing the finger-knuckleprint database. Umarani proposed a boosted geometric hashing-based technique for index table creation [133]. It extracts SIFT and SURF features and generates geometric features, which are points in a coordinate system to represent an image. The geometric hashing is boosted so that the geometric information is

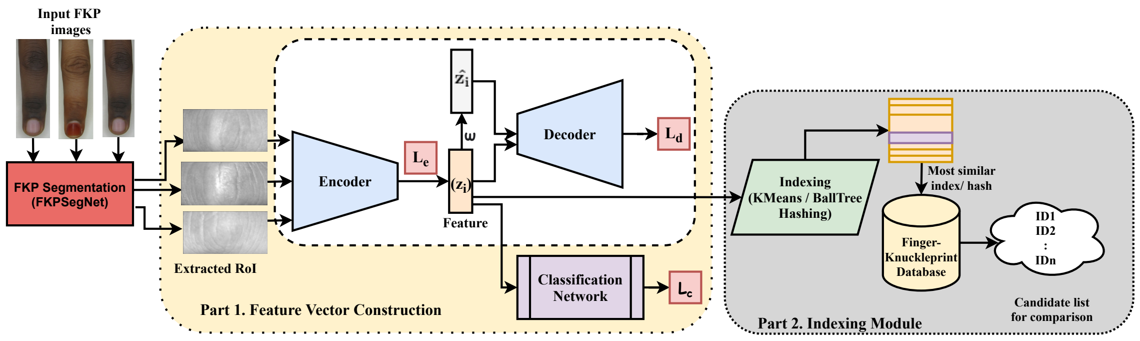


Figure 4.2: Proposed technique. Part 1 depicts the feature vector construction module (FKPIndexNet) responsible for extracting real valued 512-d latent representation. Part 2 represents indexing module that partitions the original database and associates each extracted feature with an index.

used to generate the index, and the feature descriptor is used to recognize FKP images.

4.2 Proposed Technique

This section discusses the proposed technique for indexing an FKP database. The technique consists of four main components $viz.$, Region of Interest (RoI) segmentation, feature vector construction, indexing, and retrieval. RoI segmentation aims at extracting the valuable and relevant part from the acquired image by removing the unnecessary background. The RoI and its label are given as an input to the proposed network for feature extraction. It is followed by indexing, in which each feature vector is associated with an index in the generated index table. During retrieval, the query FKP image is passed through the segmentation network for RoI extraction. A feature vector is constructed from the extracted RoI using the trained FKPIndexNet. The feature vector is compared with all the indices of the index table, and the candidates belonging to the most similar index are retrieved for comparison. All the steps are discussed in detail in the sub-sections to follow. A block diagram depicting all the stages of the proposed technique is shown in Figure 4.2.

4.2.1 FKPSegNet: Finger-Knuckleprint RoI Segmentation Network

The FKP samples can be acquired in a contactless and unconstrained environment. Acquired samples in such an environment generally contain the image of the whole finger,

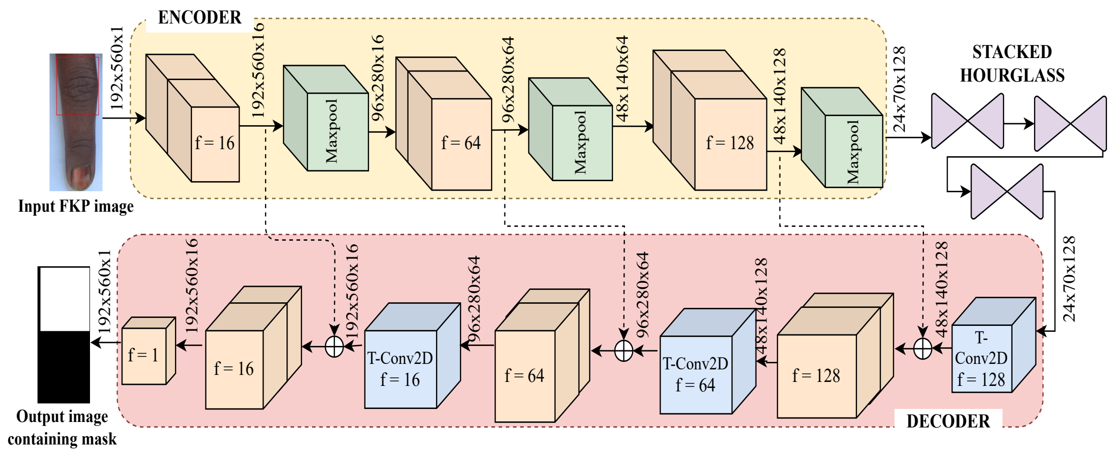


Figure 4.3: Architecture of the FKPSegNet. The orange blocks depict the pair of Convolutional layers of size 3×3 with the number of filters indicated by 'f'. The output of max-pool and transposed convolutional layer is concatenated in the decoder, using a merge connection denoted by dashed arrows.

and unnecessary background information may also get captured. Therefore, the relevant part of the image, also known as Region of Interest (RoI), containing the knuckle area with no or minimal background noise, is expected for better feature extraction. This chapter employs a novel segmentation network, FKPSegNet, to extract the desirable RoI from FKP images. The input to FKPSegNet is the acquired full finger image and a masked image depicting the desirable region of interest. The output of the network is the predicted coordinates of the RoI. The network is trained by minimizing the distance between ground truth coordinates and predicted coordinates. All the images are manually annotated to prepare the ground truth by drawing a bounding box over the knuckle area in the finger. The coordinates of the bounding box are recorded for the training of FKPSegNet. Some acquired samples with the annotated bounding boxes over the image are shown in the first row of Figure 4.1.

FKPSegNet follows the U-Net architecture [71] that consists of a contracting and an expansive path. The contracting path consists of a series of convolutional layers followed by the max-pool layer. On the other hand, the expansive path takes the output of the contracting path and puts it through a series of transposed convolutional and up-sampling layers. The high-level features from the contracting path are concatenated with the cor-

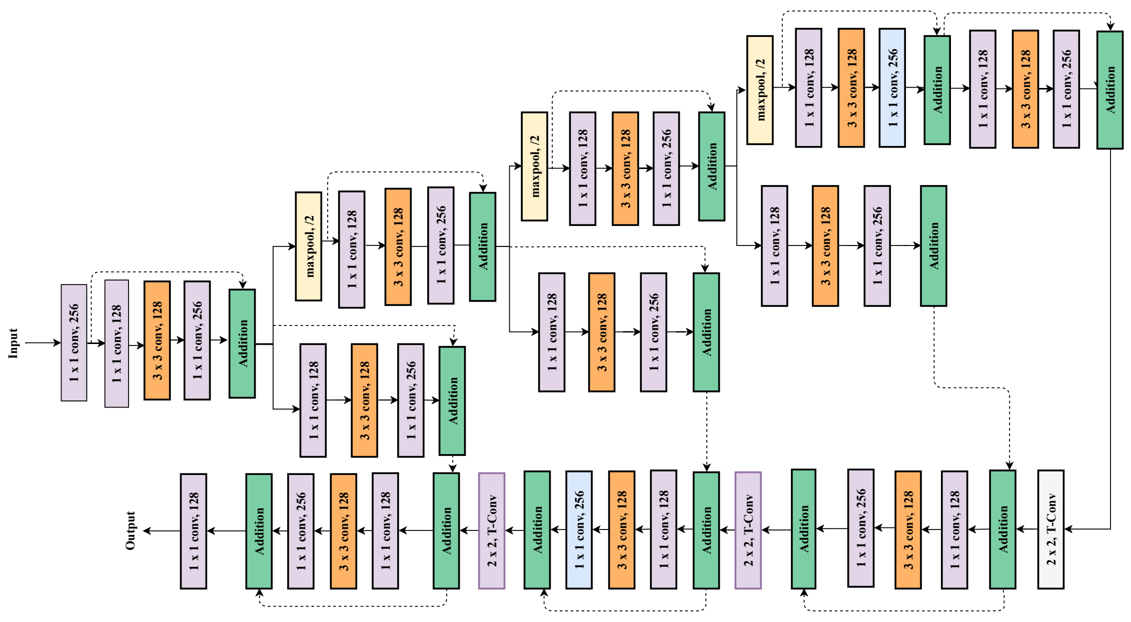


Figure 4.4: Architecture of the hourglass network (H)

responding up-sampled features in the decoder using merge connections. U-Net is commonly used in applications involving the segmentation of medical images. When FKP images are given to the standard U-Net for segmenting out the RoI, the observed output is irregular and blurry. This was due to occlusion, pose, and scale variation introduced in the images while acquiring them in an uncontrolled environment. This limitation is addressed in this chapter by proposing a stacked hourglass network between the contracting and expansive path. Hourglass network [72] includes a residual module that works on the feature vector, unlike the U-Net, which takes a complete image as an input. The residual module uses convolutional operation to learn high-level features, but it is also capable of retaining the original information using skip connections. It has a symmetric topology so that the features are extracted and consolidated across various image scales and resolutions. The output of the hourglass network is of the same size as its input feature vector. Therefore, we can say that the hourglass network only changes the depth of the data without altering its size.

FKPSegNet has three parts $viz.$, contracting path (CP), expansive path (EP) and the hourglass network (H). The contracting path (CP) aims to learn salient and discriminating features (f) corresponding to its input $i.e.$, the masked image (M). The CP consists of a pair of convolutional layers with a filter size of 3×3 and a stride of 1, followed by ReLU activation, and lastly, a max pool layer having size = 2×2 and stride=2. This block is repeated three times with different filter sizes of 16, 32, and 64. Unlike the standard U-Net, three hourglass networks are stacked over each other and are introduced in the bottleneck of U-Net for better RoI segmentation. Therefore, the output of CP is then passed to the hourglass network (H). Each hourglass consists of an encoder and decoder that has four residual modules connected in a sequential manner. The architecture of a single hourglass network is shown in Figure 5.6. After passing the feature vector f through three hourglass networks, the output is given to the EP. The expansive path consists of pair of transposed convolution layers with a filter size of 3×3 followed by an up-sampling layer. This block is repeated three times, followed by a 1×1 convolutional layer. The output of each block is concatenated with the corresponding feature map in the contracting path using a merge connection. The output of 1×1 layer in the expansive path is a probable area that is likely to contain the knuckle area or RoI. The block diagram of FKPSegNet is shown in Figure 4.3.

FKPSegNet Training and Architecture Justification. The training of the proposed FKPSegNet initiates with U-Net training without any hourglass networks in the bottleneck. As mentioned, the prediction of this network was irregular and blurred. Therefore, three stacked hourglass networks are introduced one by one in the bottleneck of the U-Net. The hourglass network compresses the feature representation to more scales to give a more precise segmentation of the RoI. The network uses long skip connections between the contracting and expansive path to facilitate easy gradient flow, which may vanish because of the deep network design. Each hourglass captures the scale, viewpoint, and occlusion invariant features. Introducing multiple hourglass networks enabled the learning of such features more effectively thus, making the RoI prediction more accurate. Therefore, to improve the segmentation capabilities, a stacked design has been used. However, using a stacked hourglass can result in a vanishing gradient problem as the network gets deeper. To avoid that, residual units have been utilized in the hourglass to allow easy flow of gradient. Since the hourglass network is very deep, it shrinks the feature vector to a very small size, which may result in the loss of spatial information. To account for this problem, long and short skip connections are used in the hourglass network. The U-Net weights with stacked hourglass are fine-tuned with the pre-trained weights of the previously trained network.

4.2.2 Feature Vector Construction

Feature vector construction is an important step before indexing of a biometric database is carried out because discriminating ability of the feature vector decides the performance of indexing and further, the identification process. The key here is that the feature vectors of the same class should be closer to each other in the latent space representation than the feature vectors belonging to different classes. This work makes use of specialized autoencoders for learning feature vectors from FKP images. Autoencoders [43] are feed-forward networks that aim at learning a feature vector representation such that when up-sampled, it is able to reconstruct the original image. It is achieved by minimizing the pixel-wise reconstruction loss that is computed between the output and the original image. However, it has been seen that the reconstruction loss does not ensure the discriminating ability of the learned representations [134]. Therefore, we propose an autoencoder based learning technique that aims not just at minimizing the reconstruction loss but also ensures that the mutual information between the input sample and its corresponding latent representation is also preserved. The proposed technique also incorporates classification loss that further ascertains that the learned feature vector has high inter-class and low intra-class dissimilarity.

The proposed technique consists of three sub-networks $viz.$ Encoder, Decoder and Classification network. The encoder takes FKP images as input and passes it through a series of convolutional layers followed by max-pool layer to learn a latent representation that has its salient and discriminating features. The latent representation is given to the decoder which up-samples them and tries to reconstruct the original image. But, as mentioned before that the reconstruction loss has no considerable impact on making the latent representations discriminating [135]. Also, reconstructing the original image from the latent representation is not sole purpose of this study. The aim is to learn such feature vectors that have high intra-class and low inter-class similarity. Therefore, some other parameters should be also be considered along with reconstruction loss to make the learned latent representations more discriminating. To do so, a relationship is established between the input FKP sample and its corresponding latent representation. This ensures that the learned latent representations preserves the discriminating features and the decoder could be used as a discriminator that tells if the latent representation is distinctive or not. Lastly, a classification network is added with the autoencoder that aims at developing a relationship between the latent representation and its label or class so that intra-class distance among samples is reduced and the intra-class distance is enhanced. This ensures that the relationship between samples in the database is effectively utilized. To understand the working of these three networks, let us assume $X = \{x_1, x_2, ..., x_n\}$ be a set of input FKP images and $Z = \{z_1, z_2, ..., z_n\}$ denotes a set of learned latent representations corresponding to the input images.

4.2.2.1 Encoder

Let E_{w_E} be the encoder that learns feature vector z_i for an FKP image x_i. As discussed earlier, the encoder in the autoencoder network learns a compressed feature representation of the input image to minimize the reconstruction loss between the input and the reconstructed image. Minimizing the reconstruction loss encourages the latent representation to retain salient representations of any input image. However, it does not fundamentally imply that the learned representation contains the optimal number of unique characteristics to the sample. Therefore, the autoencoder is regularized by considering the mutual information between the input sample and the latent representation along with the reconstruction loss in the objective function. The mutual information (MI) between x_i and z_i is maximum when the distribution of X and Z is the same. We can utilize any distribution similarity measure to compute the mutual information. In this work, we compute the mutual information between X and Z as,

$$MI(X,Z) = \int_Z \int_X p(x,z)\, \log \frac{p(x,z)}{p(x)\, p(z)}\, dx\, dz \tag{4.1}$$

where, $p(x,z)$ is the joint probability density function, and $p(x)$ and $p(z)$ is the marginal probability density function respectively. The joint probability $p(x,z)$ is equivalent to $p(z|x)\, p(x)$, where $p(x)$ and $p(z|x)$ denote distribution of input images and latent representations generated with respect to the input samples respectively. By substituting this value to Eq.(4.1) we get,

$$MI(X,Z) = \int_Z \int_X p(z|x)\, p(x)\, log \frac{p(z|x)\, p(x)}{p(x)\, p(z)} dx\, dz \tag{4.2}$$

The above equation is equivalent to computing the KL-divergence [136] between the two probability distributions $p(z|x)\, p(x)$ and $p(x)\, p(z)$. Therefore mutual information in terms of KL-divergence is as below.

$$MI(X,Z) = \text{KL}(p(z|x)\, p(x) \;||\; p(x)\, p(z)) \tag{4.3}$$

KL-divergence can take any value between zero to infinity. Zero value indicates that the two distributions are similar, while the infinity denotes entangled distributions. The upper bound of KL-divergence tends to infinity [137] especially for the MI term where the distribution of X can be unpredictable. To handle this during the optimization, a bounded measure of distribution similarity, such as JS-divergence [138], is a better choice. By replacing KL-divergence with JS-divergence, Eq.(4.3) takes the following form.

$$MI(X,Z) = JS(p(z|x)\, p(x)||p(x)\, p(z)) \tag{4.4}$$

Mathematically, the JS-divergence between any two probability distributions $a(t)$ and $b(t)$ is defined as,

$$JS(a(t)||b(t)) \;\; = \;\; \frac{1}{2}\int_T \left[a(t)\, log \frac{2\, a(t)}{a(t)+b(t)} + b(t)\, log \frac{2\, b(t)}{a(t)+b(t))} \right] dt \tag{4.5}$$

The variational estimation of JS-divergence has been proposed in [139]. It is introduced to utilize a discriminator, $D(x)$, in the autoencoder that can determine the relationship between the latent representation and the input sample. The discriminator is trained by showing a negative association of an input FKP sample with the learned representation. For that, a negative association is generated, denoted by $\hat{z}_i$, by shuffling x_i's learned latent representation z_i. This is inspired from negative sampling estimation [140] which uses a discriminator to understand the underlying distribution of the input samples. The discriminator takes a tuple containing the input FKP image and a latent representation which could be its learned one or the generated fake one. The pair of x_i and z_i is referred to as a positive pair while that of x_i and $\hat{z}_i$ is a negative pair. The discriminator is trained to differentiate between the positive and the negative pair. Let $a(t)$ and $b(t)$ denote real and fake distributions respectively, then variational estimation of Eq.(4.5) changes to,

$$JS(a(t)||b(t)) \;=\; \max_D(\mathbb{E}_{(t\sim a(t))}[log\, D(t)] + \mathbb{E}_{(t\sim b(t))}[log(1 - \; D(t))]) \tag{4.6}$$

Replacing $a(t)$ and $b(t)$ with $p(z|x)\,p(x)$ and $p(z)\,p(x)$ respectively in the above equation and substituting value of $JS(.)$ in Eq.(4.4). This loss term, given in Eq.(4.7), needs to be maximized as it represents the mutual information between the input sample and the learned latent representation. Here, the first term, $(x, z) \sim p(z|x)p(x)$, denotes that x and z form a positive pair $i.e.$ z is coming from x. While, the second term $(x, z) \sim p(z)p(x)$, indicates that they are not correlated and coming from different distributions and thus, forms a negative pair.

$$L_{MI} = max\left(\mathbb{E}_{(x,z)\sim p(z|x)\,p(x)}[log\; D(x, z)] + \mathbb{E}_{(x,z)\sim p(z)\,p(x)}[log(1 - \; D(x, z))]\right) \tag{4.7}$$

The intra-class similarity among the feature representations also plays a major role in increasing their discriminating ability. Therefore, along with maximization of the mutual information, the localization of the learned feature representations should be maximized.

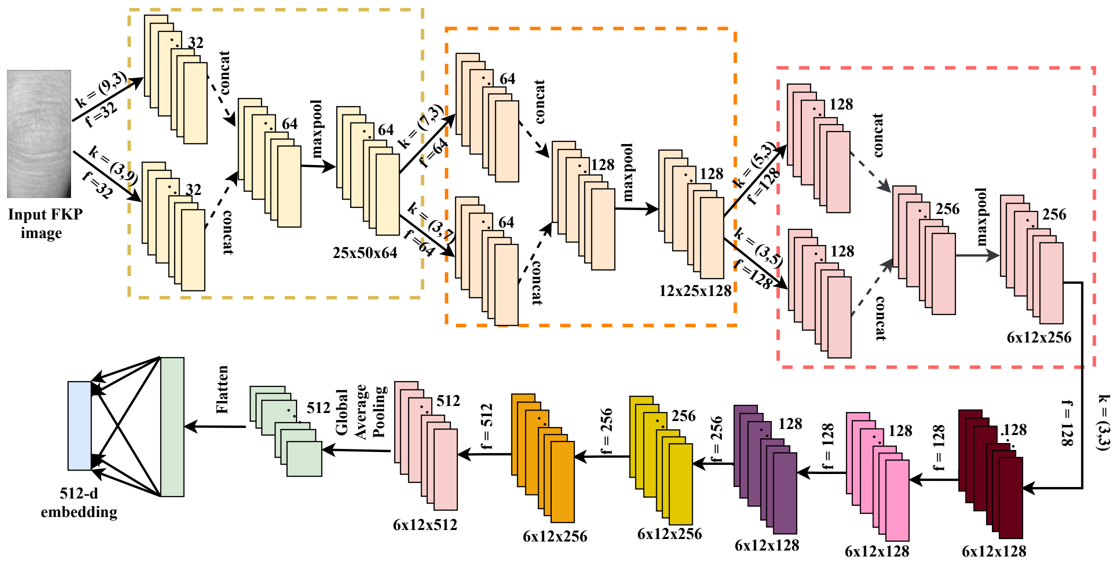


Figure 4.5: Architecture of the encoder network in FKPIndexNet

It can be achieved by making them obey a prior known distribution function, such as Gaussian distribution. Let the Gaussian distribution be denoted by $q(z)$. The distribution of feature representation space $p(z)$ is regularized by minimizing the KL-divergence between $p(z)$ and $q(z)$. Therefore, the second component of the loss function, L_{KL}, that needs to be minimized, is given as below.

$$L_{KL} = min\left(\int_Z p(z)\,\log\frac{p(z)}{q(z)}\,dz\right) \tag{4.8}$$

Both the requirements mentioned above need to be addressed by the encoder. However, the optimization objectives of both the terms is opposite. The first component, L_{MI}, that denotes mutual information between the input sample and the learned latent representation, needs to be maximized. While the second component, L_{KL}, that regularizes the latent representation space has to be minimized. The total loss function of the encoder L_E is devised by having a linear combination of L_{MI} and L_{KL}. Constant values of γ and β are used to weigh both the loss components and thus, loss function becomes,

$$\begin{aligned} L_E &= -\gamma(\mathbb{E}_{(x,z)\sim p(z|x)\,p(x)}[log\ D(x,z)] + \mathbb{E}_{(x,z)\sim p(z)\,p(x)}[log(1-\ D(x,z))]) \\ &\quad +\beta(\mathbb{E}_{z\sim p}[KL(p(z)\,||\,q(z))]) \end{aligned} \tag{4.9}$$

Encoder Architecture: The encoder in the FKPIndexNet consists of two components. The first component has three blocks having a series of convolutional layers. These layers make use of large sized asymmetric filters instead of the conventional symmetric filters. Small sized kernels produced varying activation maps and the learned feature embeddings were showing high inter-class similarity. Therefore, large sized filters are used to capture the non-rigid distortions in the FKP images. The reason of using asymmetrical filters is to facilitate the learning of line-based features found in a finger-knuckle-print sample. The lines in the FKP sample are horizontally aligned. Therefore, horizontal kernels of size 3×9, 3×7 and 3×5 are used to learn features from the knuckle lines. However, vertical filters of size 9×3, 7×3 and 5×3 are used to establish spatial relationship between the knuckle lines. The input image is passed through first convolutional layer that has one horizontal and one vertical filter of size 3×9 and 9×3 respectively. The output feature vectors from both the filters are concatenated to form a combined output. The concatenated feature vector is subject to a max-pool layer. Three such blocks with varying filter sizes are used in a sequence. After the third block, the output feature vector is capable of capturing discriminating yet salient features from the finger-knuckle-print and the distortions are no longer prominent. Therefore, this is followed by a series of convolutional layers with smaller filter size and lastly, a global average pooling layer. The output of global average pooling layer is flattened to output a 512-d feature vector that best represents the input FKP sample. This feature vector is fed to the decoder. The architectural diagram of the encoder network is shown in Figure 4.5.

4.2.2.2 Decoder

The aim of the decoder network is to reconstruct the original image corresponding to the input latent representation. The input to the decoder network is the latent representation

($z_i \in Z$) and the output is a reconstructed image ($\hat{x_i}$). The network is trained to minimize the pixel-wise mean squared error, also known as the reconstruction loss, between x_i and $\hat{x_i}$ over all the image samples. If there are N number of samples in the training set, the reconstruction loss (L_r) is defined as given in Eq.(4.10).

$$\begin{aligned} L_r &= \frac{1}{N}\sum_{i=1}^{N} \|D_{w_D}(E_{w_E}(x_i)) - x_i\|_2^2 \\ &= \frac{1}{N}\sum_{i=1}^{N} \|\hat{x_i} - x_i\|_2^2 \end{aligned} \tag{4.10}$$

where, w_D and w_E are the weights of decoder and encoder respectively and $||.||_2^2$ denotes the euclidean distance between both the inputs. The reconstruction loss depends on two factors which are, distribution of the latent representations and the reconstruction ability of the decoder network. However, the discriminative ability of the features is more important for indexing than their reconstruction ability. It has been seen that more often the data contains nuisance factors such as, illumination and occlusion in the FKP images. These factors are not relevant in the prediction process but they may interfere with the feature extraction process. This may result in the less efficient feature embeddings because of capturing unnecessary information which could be in the form of associating illumination with a class of images etc. One solution to this problem is to make the network learn a set of such factors that are irrelevant to the prediction. This can be done by training the network on augmented dataset that covers all the irrelevant information. This is a good solution but it would make the network robust to only the seen variations which would be limited in number. Therefore, the network performs poorly on the data containing unseen nuisances. Another solution is to train the network to get rid of the nuisance factors from the learned latent representations [141]. Models trained in this way become robust by exclusion rather than inclusion thus, performing well even on the unseen nuisances. Let f_{x_i} be a set of features that defines the predictability of x_i and $\hat{f_{x_i}}$ denotes the set that contains irrelevant features. Then, the latent representation (z_i) contains all the information that is required to predict the class of x_i and $\hat{z_i}$ contains all the irrelevant information which is not presented to the system thus, avoiding learning of inaccurate associations.

A noisy transformer (ω) is incorporated in the autoencoder network to generate a noisy latent representation ($\hat{z}_i$), corresponding to each z_i. Both the representations are then passed to the decoder network that constructs x_{z_i} and $x_{\hat{z}_i}$. The decoder network is trained by minimizing the relative reconstruction loss between both the outputs. Along with this, the standard reconstruction loss is also minimized between x_i and x_{z_i} to ensure the correct working of the decoder network. The total loss of the decoder can be written as given in Eq.(4.11).

$$L_D = \frac{1}{N}\sum_{i=1}^{N}\Big(\|x_{\hat{z}_i} - x_{z_i}\|_2^2 + \|x_{z_i} - x_i\|_2^2 \Big) \tag{4.11}$$

The autoencoder is trained by minimizing the summation of L_E, reconstruction loss and relative reconstruction loss as explained above. Let w_E and w_D be the parameters of encoder and decoder respectively, the total loss of autoencoder network L_{ae} can be defined as:

$$L_{ae} = \underset{w_E, w_D}{min}\,(L_E + L_D) \tag{4.12}$$

4.2.2.3 Classification Network

The loss function L_{ae} results in learning of feature embeddings for FKP samples that have high inter-class dissimilarity. However, this does not ensure that the intra-class similarity among the embeddings is maximized. The classification network is utilized in order to reduce the intra-class distance among the latent representations. This is achieved by associating the latent representation with a class $i.e.$ labelling a latent representation with a subject ID to which it belongs. After training the autoencoder to minimize the loss given in Eq.(4.12), the parameters w_E and w_D are taken as initial parameters for the classification network. Further, the classification network is trained with the parameters of the autoencoder. The classification loss can be written in terms of latent representation by modifying the last term in Eq.(4.9) as,

$$L_C == E_{x \sim p(x)}[KL(p(l,z)|x)||q(l,z)] \tag{4.13}$$

where l is the label of the feature vector z, $p((l,z)|x) = p(l|z)p(z|x)$ and $q(l,z) = q(z|l)q(l)$. It should be noted that the distribution $q(z|l)$ is a normal distribution with mean and variance being equal to μ_l and 1 respectively. The three networks are combined and trained together in an end-to-end manner by minimizing the linear combination of L_E, L_D and L_C respectively.

$$L_{FKPIndexNet} = \underset{w_E, w_D, w_C}{min} (L_E + L_D + L_C) \tag{4.14}$$

4.2.3 Indexing

In the proposed technique, each FKP image is represented by a feature vector of fixed length extracted from the proposed network (FKPIndexNet). The indexing phase associates the extracted feature vectors to an index and stores them with their similar samples in an index table. To create the index table IT for an FKP database, three techniques $viz.$ k-means clustering [142], BallTree hashing [143] and Locality Sensitive Hashing [115] have been explored and implemented on the obtained feature vectors set. These techniques would group the samples so that the similar ones stay together in the same group, and the dissimilar ones would lie in different groups. k-means outputs clusters while BallTree and locality sensitive hashing partitions the search space. Therefore, the cluster centers or the hash of a group would serve as the index and all the samples belonging to that index would lie in the bucket of that index. All three indexing techniques are explained below. Algorithm 5 shows the pseudo-code for the indexing process.

1. **k-means Clustering:** The objective of k-means clustering algorithm is to partition a set of feature vectors, $f_{x_1}, f_{x_2}, ..., f_{x_N}$ into 'k' disjoint groups $c_1, c_2, ..., c_k$. It brings the similar feature vectors together while separating those that are different. Each group or cluster would have a representative data point also known

Algorithm 5 Indexing the extracted feature vectors

```
Require: Embedded feature space $F$
Ensure: Index table $IT$
1: for every FKP image $x_i$ do
2:     Extract feature $f_{x_i} = e_{w_e}(x_i)$ using the trained network
3:     Append $f_{x_i}$ to $F$
4: Apply clustering (k-means) or hashing (BallTree/LSH) on embedded feature space
   $F$ till convergence
5: for each cluster cente (k-means) or hash value (BallTree/LSH) do
6:     Create an entry in index table $IT$.
7:     Put ID of all FKP samples with their corresponding feature vectors, lying in the
   cluster or hash, in the bucket of $IT$
8: return $IT$
```

as mean of all the feature vectors. The algorithm starts by initializing 'k' centers using k-means++ initialization. k-means is a distance-based clustering and therefore, computes euclidean distance of each feature vector f_{x_i} from all the centers $m_1, m_2, ..., m_k$. A point f_{x_i} is assigned the closest cluster $i.e.$ one with the least euclidean distance. The cluster centers are calculated by finding the mean of all the feature vectors assigned to it and it is updated after every iteration. The error is computed in each iteration using Eq.(4.15). This is repeated till no further change in cluster assignment is observed or maximum number of iterations have been exhausted. After convergence, the cluster centers are stored as hash values in the index table and the FKP IDs along with their feature vectors that lie in a particular cluster are stored corresponding to it.

$$E_{k-means} = \sum_{j=1}^{k} \sum_{(f_{x_i}) \epsilon c_j} ||f_{x_i} - m_j|| \tag{4.15}$$

2. **BallTree Hashing:** BallTree is a space-partitioning algorithm that divides the data points in hyperspheres or 2-D circular space. A partition is represented by center ($center_j$) and diameter (d_j) of the circular segment. A data point (x_i) is said to be belonging to a segment (s_j) if the euclidean distance between x_i and $center_j$ is less than the radius $i.e.$ $||x_i - center_j|| < \frac{d_j}{2}$. Graphically, a ball-tree is represented as

a binary tree in which each node depicts a partition. Considering two nodes n_1 and n_2, if n_1 is child of n_2, then the segment s_2 is sub-segment of s_1. Initially, ball-tree contains just one node and all the data points are assigned to this node. After that, partitioning of tree is initiated by following a divide and conquer approach. For a segment (s_j), the partitioning procedure works as follows.

(a) The farthest data point from $center_j$ is selected and is assigned to the left node of s_j. It is denoted by ${s_j}^L$.

(b) Now, the farthest data point from ${s_j}^L$ is found out and it becomes right child $({s_j}^R)$ of s_j.

(c) The data points in s_j are assigned to either of the left and right segment based on which of ${s_j}^L$ or ${s_j}^R$ is closer to the considered data point.

(d) These sub-partitions are allocated to child nodes of n_j, ${n_j}^L$ and ${n_j}^R$ respectively.

3. **Locality Sensitive Hashing (LSH):** A Locality Sensitive Hashing (LSH) function maps the feature vectors to a lower-dimensional representation such that the feature vectors that are similar to each other are mapped in the same bucket with a high probability in the lower-dimensional space. The main objective of LSH is to maximize the probability of collision of similar items $i.e$, the probability of two similar feature vectors lying in the same bucket should be high. The hash function for an input feature vector f_{x_i} is computed by using two random values, $\vec{r}$ and u. Here, r is a d-dimensional vector whose entries are randomly chosen from a set of vectors following the Gaussian distribution. The dot product is quantized into a set of hash bins with the objective that all the nearby feature vectors should lie in the same bucket as shown,

$$h^{r,u}(f_{x_i}) = \left\lfloor \frac{\vec{r} \cdot f_{x_i} + u}{w} \right\rfloor \tag{4.16}$$

In this equation, w is the quantization width and u is a random variable lying between 0 and w. Quantization width determines the number of entries or candidates that would lie in each bucket of the hash table. Increasing the quantization width results in compact table as each bucket will have more number of entries. On the other hand, lower value of w results in larger table with lesser number of candidates in each bucket of the hash table. The search for true match for the query image is accomplished in a linear manner $i.e.$, the query image is compared with all the candidates lying in the selected bucket of the index table. Therefore, there is a trade-off between the table size and final number of comparisons. Two conditions must be satisfied to serve the purpose of reducing number of comparisons for identification of a query FKP sample. These are,

- The probability of two feature vectors lying in the same bucket of index table should be high if they are close to each other in the feature embedding space. Let there are two feature vectors represented by f_{x_1} and f_{x_2}. Let the euclidean distance between the two feature vectors is $< d_1$. This distance is $\leq d_1$ which is the threshold distance value that determines if the given two feature vectors are close to each other in the feature embedding space. In this case, both f_{x_1} and f_{x_2} will lie in the same bucket. This is mathematically represented as,

$$P[h(f_{x_1}) = h(f_{x_2})] \geq p_1 \text{ if } \|f_{x_1}, f_{x_2}\| = d_1 \leq d \tag{4.17}$$

- Contrary to the previous condition, this condition states that the probability of two dis-similar feature vectors, f_{x_1} and f_{x_3}, lying in the same bucket should be low. Let d_1 be the euclidean distance between f_{x_1} and f_{x_2} and d_2 is the euclidean distance between f_{x_1} and f_{x_3}. Since f_{x_1} and f_{x_3} are dis-similar feature vectors, the distance between them should be greater than d $i.e.$, $d_2 \geq a \times d$, where a is any constant. Therefore, the probability of them lying in the same bucket of the index table will be less. Mathematically, it can be shown

Algorithm 6 Retrieval IT, q_i

```
Require: $IT$: Index table, $Q$: probe FKP images set
1: for each query FKP sample $q_i \in Q$ do
2:     Extract the latent representation, $f_{q_i}$
3:     for each bucket representative in $IT$ do
4:         Compute cosine similarity between hash value and $f_{q_i}$.
5:         Create a table $S$ with index and its corresponding cosine similarity score with $f_{q_i}$.
6:     Find the maximum score value in $S$.
7:     Retrieve IDs stored in the selected bin to get candidate list $C$ for matching.
8:     return C
```

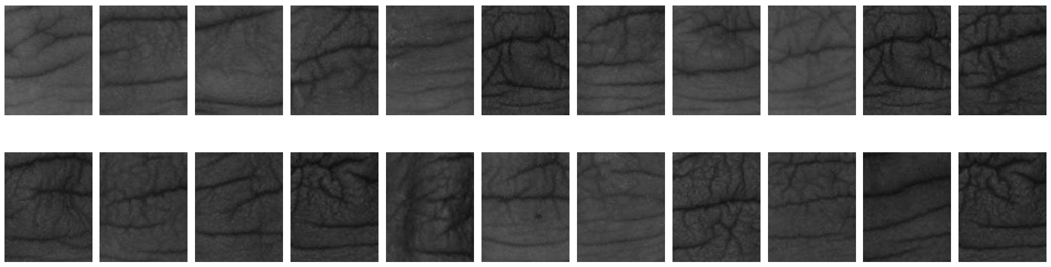


Figure 4.6: Segmented finger-knuckleprint images taken from IITD FKP database.

as,

$$P[h(f_{x_1}) = h(f_{x_3})] \leq p_1 \; for \; \|f_{x_1}, f_{x_3}\| = d_2 \geq a \times d \tag{4.18}$$

To further increase or reduce the probability given in the conditions respectively, a hash function of t- bits can be generated by performing t dot products in parallel using Eq.(4.16). A $t-$bit hash value of a feature vector belonging to FKP sample x_1 can be computed by concatenating t values determined using the Eq.(4.16). $h(f^t_{x_1})$ can be written as $= h_1(f_{x_1}), h_2(f_{x_1}), \ldots, h_t(f_{x_1})$. After implementing LSH on the set of feature vectors generated by the trained model, we get a data structure that consists of hash value and the candidate IDs in its corresponding bucket.

4.2.4 Retrieval

The output of the indexing component is an index table, with each feature vector being associated with an index in the table. The table is created such that similar feature vectors tend to fall in the same bucket, and different ones lie in different buckets. The objective of the retrieval stage is to output a suitable list of candidates for comparison with the probe FKP image. The output candidate list is expected to be small compared to the database's size for an efficient identification process. When a query FKP image, q_i, is shown to the identification system, it is fed to the FKPIndexNet for feature extraction. The obtained feature vector, denoted by, f_{q_i}, is compared with every index in IT. The index table size equals the number of clusters in k-means clustering or the number of partitions in BallTree or locality-sensitive hashing. The index having maximum similarity with the probe feature vector is selected for candidate set generation. Hence, all the candidates' IDs and their feature vectors lying in the selected index are retrieved for similarity computation with the probe image feature vector. A score list (S) consisting of a similarity score for q_i with the candidates in the list is formed. S is sorted in descending order, and the rank of q_i's true match is obtained. A probe's true match may not be found in the selected index. To handle this issue, the retrieval algorithm will refer to the neighboring buckets in the index table. The next best cluster center or partition will be selected, and its candidates are fetched for similarity computation with the probe image. It is repeated till a true match is found. The size of the retrieved candidate list is fixed, making the identification a constant time operation.

4.3 Experimental Results

This section details the experimental setting, such as the databases' specifications and an overview of the training and testing protocol. The proposed technique is evaluated for both the recognition and identification process. First, the extracted features are used for evaluating the verification system, followed by identification performance after the database was indexed using the proposed technique.

4.3.1 Database

To validate the performance of proposed technique for indexing FKP datasets, experiments are conducted on two publicly available datasets $viz.$ Hong Kong PolyU Finger

Table 4.1: Database specification with their recognition performance achieved by the proposed technique.

Parameters	**PolyU-FKP [53]**	**IITD FKP [54]**
No. of subjects	660	158
Gallery samples	3960	316
Probe samples	3960	473
Genuine comparisons	23763	948
Imposter comparisons	15661794	148836
Total comparisons	15685557	149784
Equal Error Rate	2.76%	1.72%
Accuracy	97.25%	93.67%
Discriminative Index	2.68	3.27

Knuckle Print (PolyU-FKP) [53, 144] and IIT Delhi Finger Knuckle database (IITD FKP) [54, 124]. Details about the two databases are provided below. Some sample images from PolyU-FKP database and IITD FKP are shown in Figure 4.1 (first row) and Figure 4.6 respectively.

1. **PolyU-FKP [144]:** Hong Kong PolyU Finger Knuckle Print database (PolyU-FKP) is a large scale knuckle image database that has been acquired in contactless setup using a simple hand-held camera. There are 503 subjects, out of which some have provided FKP samples from both the hands resulting in 660 classes. This work considers twelve FKP images per class and utilized first six for training and the remaining for testing. Therefore, in total there are 7,920 FKP images in this database. All the images are in bitmap format and the resolution of RoI samples is 50×100 pixels.

2. **IITD FKP [54]:** IITD FKP Database consists of 790 FKP images collected from 158 individuals between August 2006 and June 2007. All the subjects are aged between 16 to 55 years. The images are in bitmap (.bmp) format and have resolution of 80×100 pixels.

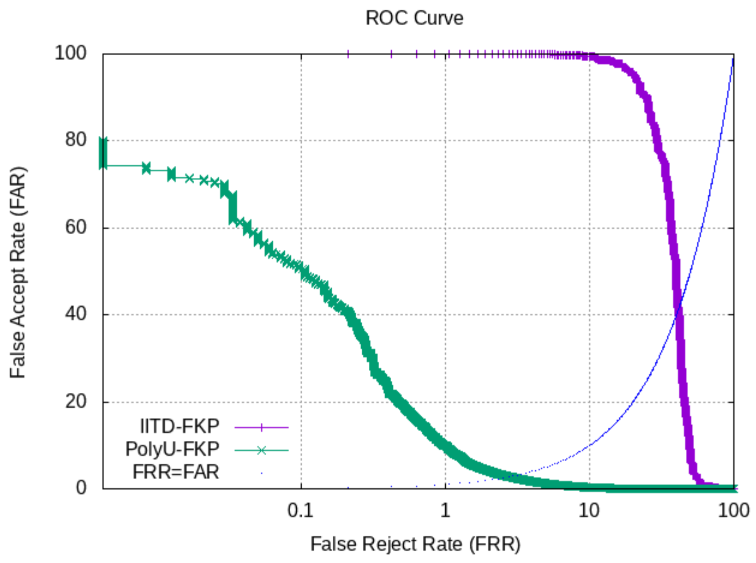


Figure 4.7: Receiver Operating Characteristic (RoC) Curve of the proposed technique (plotted on log scale) on the considered datasets.

4.3.2 Testing Protocol

Different training-testing strategies are employed for FKP identification since both the databases contain different number of images. For PolyU-FKP database, images collected during the first session $i.e.$ first six images are used for training the network while the remaining six images are used as query images for testing. On the other hand, first two images from the IITD FKP database are used for training and remaining 3 images are used as probe samples. The details regarding the training and testing split is also given in Table 4.1.

4.3.3 Results

To validate the performance of the proposed approach, results are computed for both verification and identification system. Firstly, the quality of the learned features are computed because it is their discriminative ability that determines the performance of the indexing module. Therefore, the recognition results are listed followed by indexing performance based on two different techniques namely, k-means clustering, BallTree hashing and Locality Sensitive Hashing (LSH).

Table 4.2: Penetration rates (%) at different hit rate (HR) values when BallTree and LSH have been applied on PolyU-FKP database and IITD FKP database (Lower is better).

BallTree Hashing			Locality Sensitive Hashing		
HR (%)	**PolyU-FKP**	**IITD FKP**	**HR (%)**	**PolyU-FKP**	**IITD FKP**
70	15.70	46.20	70	-	-
75	20.50	49.78	75	-	-
80	26.13	54.00	80	0.025	-
85	33.96	57.17	85	0.05	-
90	44.92	60.33	90	0.23	-
95	63.18	62.86	95	1.21	-
100	98.96	66.45	100	3.42	0.316

4.3.3.1 Recognition Performance

To evaluate the verification performance of the proposed approach, Accuracy, EER, and DI of the system are computed. These are computed by comparing each sample in the test partition with all the samples in the training partition. The details regarding the number of comparisons involving genuine and imposter comparisons are given in Table 4.1. In the table, gallery images refer to training split as these are the images that are indexed. In contrast, probe samples refer to the testing split as these images are used as query images for identification. The proposed technique achieved Accuracy = 97.25%, EER = 2.76% and DI = 2.68 on PolyU-FKP and Accuracy=93.67%, EER = 1.72% and DI = 3.27 on IITD FKP dataset. Although the verification performance is not the main aim of this study, the proposed technique achieves high accuracy with low EER thus, indicating high discriminating and representative ability of the learned feature vectors. The ROC Curve of the recognition system using the features extracted by the proposed technique on all the considered datasets is shown in Figure 4.7.

4.3.3.2 Indexing Performance

Indexing performance of the proposed technique on PolyU-FKP and IITD FKP database has been evaluated in terms of hit rate and penetration rate. A high value of hit rate with lower value of penetration rate is expected for a good indexing technique. k-means clustering, BallTree hashing and LSH have been utilized for index table creation. BallTree hashing performs poorly as compared to other two techniques. It achieves 63.18% and

Table 4.3: Penetration rates (%) at different hit rate (HR) for when k-means clustering is used for indexing features of PolyU-FKP database and IITD FKP database (Lower is better).

PolyU-FKP					IITD FKP				
HR	**k=5**	**k=25**	**k=65**	**k=85**	**HR (%)**	**k=5**	**k=25**	**k=65**	**k=85**
70	0.90	4.59	**2.97**	3.10	**70**	11.07	12.02	**11.39**	12.02
75	1.79	5.60	**3.88**	3.90	**75**	13.92	14.24	**13.92**	14.24
80	5.37	7.19	**5.40**	5.05	**80**	18.98	16.45	**16.45**	17.08
85	16.74	10.22	**7.77**	7.22	**85**	21.83	20.25	**20.25**	19.93
90	23.81	14.89	**11.64**	10.83	**90**	25.63	24.36	**25.31**	24.68
95	38.81	23.61	**21.81**	19.45	**95**	37.02	34.17	**34.17**	34.17
100	89.97	95.75	**85.30**	86.21	**100**	60.44	57.59	**57.91**	58.86

62.86% penetration rate at 95% hit rate on PolyU-FKP and IITD FKP database respectively. The results are shown in Table 4.2. In k-means clustering, the number of clusters have been varied between 5 to 85 with a difference of 10 to observe the effect of cluster number on hit rate and penetration rate. Hit rate and penetration rate w.r.t. different number of clusters, when k-means has been implemented for indexing, are reported in Table 4.3. It can be observed that the optimal value of penetration rates for different hit rate can be attained when number of clusters were fixed at 65. After this value, the penetration rate again started to increase. It can be observed that to get 100% hit rate, only 85.30% and 57.91% of the PolyU-FKP and IITD FKP database needs to be retrieved. FKPIndexNet with LSH gives the best performance among the three techniques. When index table is created using LSH, a penetration rate of 3.42% and 0.32% for PolyU-FKP and IITD FKP database respectively is required at 100% hit rate. This means only 3.42% and 0.32% of the considered databases are required to be 100% sure that the true match can be found in the retrieved candidate list. The results have been documented in a table for both the datasets as shown in Table 4.2.

A graph depicting hit rate vs. penetration rate relationship for all the three techniques corresponding to both the datasets is shown in Figure 4.8. The graph clearly shows that LSH performs better than the remaining two approaches and attains a very low penetration rate on both the datasets. Therefore, LSH is selected for index table creation and

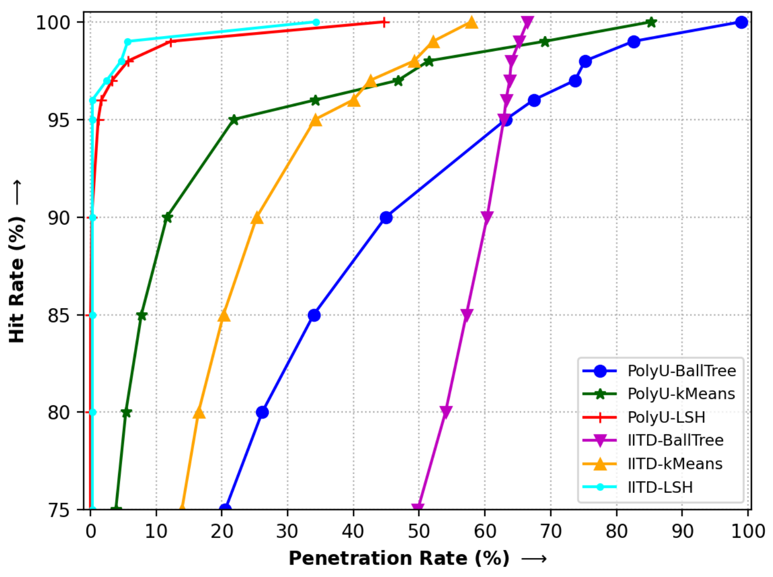
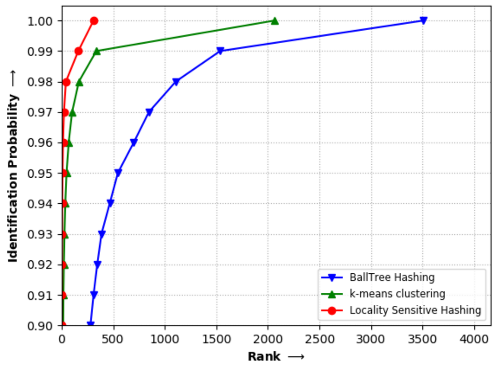


Figure 4.8: Graph showing hit rate (%) vs. penetration rate (%) achieved by different indexing techniques on the considered databases.

further, for identification process. The performance of closed set identification system can be summarized using CMC. The CMCs are plotted for PolyU-FKP and IITD FKP databases and is shown in Figure 4.9(a) and Figure 4.9(b) respectively. The graphs portray the relationship between identification probability obtained at various ranks by the proposed technique. It is evident from the graphs that LSH gives better identification probability than k-means clustering and BallTree hashing when the features are extracted using the proposed technique. Rank-1 identification rate was also computed to determine the number of correctly identified queries at the top most rank. It was found out that rank-1 identification rate of PolyU-FKP database was 61.9%, 17.65% and 89.69% when BallTree hashing, k-means clustering and LSH respectively were implemented. However, the obtained values on IITD-FKP was 1.2%, 16.45% and 17.41% respectively. Therefore, we can conclude that the maximum number of queries were correctly identified at rank-1 when FKPIndexNet with LSH is implemented to index the considered databases. The proposed indexing technique generates a fixed size candidate list. Therefore, the time required for identification becomes constant.



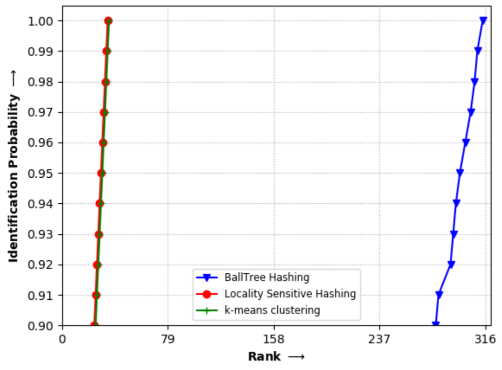


Figure 4.9: CMC showing relationship between identification probability and rank for PolyU-FKP and IITD FKP datasets when identification is done using the proposed indexing technique.

Indexing Performance Comparative Analysis: A geometric hashing based technique that uses SIFT features for indexing FKP database has been proposed by Jayaraman et al. [133]. This technique has used a different training-testing protocol $i.e.$ 11 images of each subject have been used for training and the last image has been used as the probe image. The comparison between both the techniques is shown in Table **??**. The values in the cells refers to the penetration rate for different values of Hit Rate for both the techniques. It can be seen that even though the proposed technique (FKPIndexNet) uses a more strict training-testing protocol, it performs better than the state-of-the-art technique. The proposed technique with LSH requires 3.33% penetration rate at 97% hit rate on PolyU-FKP database. While, it has been seen that geometric hashing requires 40.41% of the database for 97% hit rate. This is 12 times less database that needs to be seen for 97% hit rate. The time required for identification is also an important measure to gauge the performance of the proposed method. To do so, the retrieval time has been computed in two scenarios- without indexing and with indexing the database. It has been observed that the time required for identification for a probe image Q_i with indexing is approximately 21 times lesser than the time required using the linear search.

Time analysis: A time-based analysis is conducted to evaluate the performance of the identification process while using the indexed database and with exhaustive search, in

Table 4.4: Comparison of penetration rate (%) at specific hit rates with existing work on PolyU-FKP database. (Lower is better)

Technique / Hit Rate	**97%**	**99%**	**100%**
Database: PolyU-FKP			
BallTree hashing [143]	74.01%	90.17%	98.96%
Geometric Hashing (SIFT) [133]	40.41%	94.07%	-
k-means clustering [142]	32.56%	67.03%	85.3%
Proposed: FKPIndexNet	2.46%	2.91%	3.42%
Database: IITD-FKP			
BallTree hashing [143]	65.61%	66.03%	66.45%
k-means clustering [142]	43.67%	46.83%	49.05%
Proposed: FKPIndexNet	0.32%	0.32%	0.32%

which the query sample is compared with all the templates of the database. The comparison is done in terms of speedup to determine the reduction in identification time while using generated candidate list over the complete database. Running time is computed for only the retrieval stage wherein the suitable candidates are retrieved for comparison with the query image. The time required for RoI segmentation, pre-processing and feature extraction are common even for exhaustive search. Additionally, the index table generation is an offline and one-time process and therefore, time for these components are not taken into account for computing the speedup. The proposed technique requires 168.98 seconds and 2.16 seconds when identification is performed with the naive approach i.e. exhaustive search on PolyU-FKP and IITD FKP database respectively. However, this time reduced to only 14.64 seconds and 0.47 seconds, respectively, when the considered FKP databases are indexed using FKPIndexNet with LSH. BallTree hashing and k-means clustering required 18.76 and 0.54 seconds on PolyU-FKP and 20.31 and 0.80 seconds on IITD-FKP database respectively. Table 4.5 shows a comparison of the identification time taken for a query FKP with and without indexing on the considered databases. It can be observed that a speedup of 11 and 4 times is obtained when identification is done using the indexed PolyU-FKP and IITD FKP database respectively.

Table 4.5: Time required (in seconds) for the query FKP identification using the proposed indexing technique and exhaustive search (PC comparison).

Database	Non-Indexed	k-means [142]	BallTree [143]	LSH [115]
PolyU-FKP [53]	168.98	20.31	18.76	14.64
IITD FKP [54]	2.16	0.80	0.54	0.47

4.3.3.3 Ablation Study

The proposed technique generates an index table using the learned features of the FKP images. The quality of extracted features determines the performance of any biometric indexing technique during the identification process. Therefore, an ablation study is done to analyze the effect of varying feature dimension on hit rate and penetration rate. Feature vectors having three different dimensions 128, 256 and 512 have been analyzed to determine the suitable feature vector dimension for indexing the FKP databases. Subsequently, separate index tables are generated for the three sets of feature vectors. Penetration rate is computed for different values of hit rate for the query images from PolyU-FKP database. The obtained result is shown in Table 4.6. Hit rate refers to the confidence with which true match of the query image can be found out from the retrieved subset of the database. On the other hand, penetration rate determines the percentage of the database required to find true match of a query sample. A good indexing technique is expected to achieve high hit rate from a small list of candidates $i.e.$ at lower value of penetration rate. It can be empirically determined from the Table 4.6 that the 512-d performs best on the PolyU-FKP database. Therefore, 512-d feature vectors are used throughout the experimentation.

4.4 Summary

This chapter proposes a novel technique for indexing FKP databases that learns a fixed-length and discriminative feature vector for FKP images. The first part of the chapter proposes a novel segmentation network, FKPSegNet, to extract the Region-of-Interest (knuckle region) from an acquired finger-dorsal image. It is a U-Net based network that employs three stacked hourglass networks in the bottleneck. Later, a specialized autoencoder network that learns feature embeddings from the FKP images has been proposed. It

Table 4.6: Ablation study showing the effect of feature dimension on hit rate and penetration rate for PolyU-FKP database. The cell values denote penetration rate(%) at various hit rate.

Indexing	**Feature**	**Hit Rate**						
		70%	**75%**	**80%**	**85%**	**90%**	**95%**	**100%**
k-means	128-d	4.14	5.43	7.32	11.01	16.59	28.16	97.55
	256-d	4.7	6.14	8.36	11.52	17.22	29.55	96.67
	512-d	2.97	3.88	5.4	7.77	11.64	21.81	85.3
BallTree	128-d	14.26	21.37	30.58	39.44	58.63	67.21	99.15
	256-d	15.7	20.5	26.13	33.96	44.92	63.18	98.96
	512-d	9.14	11.99	16.33	22.47	31.64	48.25	97.57
LSH	128-d	-	-	-	-	0.02	14.57	56.98
	256-d	-	-	-	-	-	0.02	44.67
	512-d	-	-	-	-	-	-	0.02

is trained by using a novel custom loss function which minimizes the reconstruction loss and maximizes the mutual information between the input sample and its learned latent representation. It also incorporates classification loss to increase intra-class compactness among the feature vectors. The encoder in the proposed feature extraction network employs asymmetric filters instead of symmetric ones to learn line-based FKP features and establish a spatial relationship among them. Three different techniques $viz.$ k-means Clustering, BallTree hashing and LSH have been explored for index table generation. Experiments are performed for verification and identification of the proposed technique on two publicly available benchmark FKP databases $viz.$ PolyU-FKP and IITD FKP. The proposed technique achieved 100% of hit rate at the penetration rate of only 3.42% and 0.32% for PolyU-FKP and IITD FKP database respectively. Hence, by using FKPIndexNet, we need to search only a small percentage of the database instead of the whole database for identification and that too without compromising on the accuracy. The next chapter addresses indexing of fingerprint database.

Chapter 5

Fingerprint Indexing

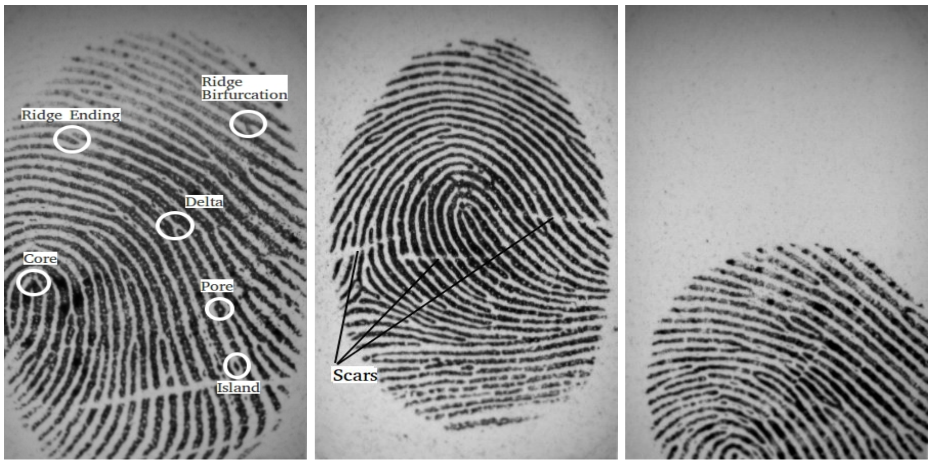
Fingerprint [3] is a skin pattern acquired from the tip of a finger. It is a widely accepted biometric trait for access control because of several advantages in terms of uniqueness, temporally invariant, easy acquisition, and acceptability among the population. The curves present on a fingerprint, called ridge lines, form unique patterns in every fingerprint. A fingerprint is a quite expressible biometric trait as there exists a considerable amount of discerning patterns such as loops, whorls, and arches on it [145]. A ridge line either terminates or bifurcates in two ridge lines. Minutiae points that refer to the irregularities in the ridge structure $i.e.$ the points of termination and bifurcation of the ridge line. A minutiae point is represented as a tuple containing three values $viz.$ its (x, y) coordinates and its orientation in degrees. The orientation of the detected minutiae point is determined by the direction of the ridge line. A fingerprint is unique among the population as the probability of two fingerprints being identical is as low as 1.5×10^{-15} [18]. A fingerprint is easy to acquire and requires less user cooperation.

The Automatic Fingerprint Identification System (AFIS) identifies humans based on their fingerprints. Fingerprint comparison algorithms that run behind AFIS works accurately and speedily for small databases [146]. However, as the size of the fingerprint database grows, identification becomes computationally expensive, and despite taking considerable time and memory, AFIS is unable to provide an accurate result [147]. In order to reduce the time taken in identification, the search space needs to be reduced to lower the number of candidates for comparison without compromising on the accuracy of identification. It is achievable by indexing the fingerprint databases. The process starts by constructing a feature vector that consists of a fingerprint's essential yet distinguishing characteristics. Each feature vector is then associated with an index that closely resembles it. An index table is generated by putting similar feature vectors in the same index. Retrieval works by finding the most similar index for the query sample's feature vector and fetching the feature vectors lying in that index for comparison. The underlying idea is to highlight a few fingerprints from the database to compare the query image with only those fingerprints.

This chapter presents an effective technique to index fingerprint databases. The proposed indexing technique encodes the spatial and directional relationship between the minutiae points and the core point. Therefore, the initial step is to locate the core and minutiae points in the fingerprints. A novel U-Net-based deep learning architecture with a stacked hourglass is proposed for core point detection from fingerprint images. The minutiae points are extracted using a minutiae detector, called MINDTCT, contained in NIST Biometric Image Software [148]. A feature vector is constructed by learning the relationship between each minutiae point and core point. This feature vector is used for index table generation. The rest of the chapter is organized as follows. Section 5.1 presents a literature review of the fingerprint indexing techniques. Section 5.2 discusses the proposed technique for core point detection and indexing. Results are discussed in Section 5.3.

5.1 Literature Survey

Fingerprint indexing techniques differ based on the features used to construct the index table, such as core, delta, ridge endings, ridge bifurcations etc. Different fingerprint features are shown in Figure 5.1 (a). The literature review of fingerprint techniques is classified into four categories $viz.$ texture-based, minutiae-based, hybrid, and deep learning



(a) Fingerprint Features (b) Scar on fingerprint (c) Partial fingerprint

Figure 5.1: Figures showing: (a) fingerprint features, (b) scarred fingerprint and (c) partial fingerprint with missing core point

based techniques depending on the type of feature used for the index table generation.

5.1.1 Texture Based Indexing Techniques

These indexing techniques involve using global features such as ridge orientation field, ridge frequency field, ridge pattern types, ridge flow structure, core, and delta points for index table construction. The global features portray fingerprints in a global view. An indexing approach based on scalar and vector features acquired from ridge-line orientation and frequency has been proposed in [149]. The technique has been tested over six databases: NIST DB4, NIST DB4 (natural), NIST DB14, FVC2000 DB2, FVC2002 DB3, and FVC2002 DB1. In [150], a clustering-based indexing technique in which polar complex moments (PCMs) were employed to construct feature vectors has been proposed. The extracted fingerprint representation is rotation invariant. The proposed approach has been tested on FVC2002 DB1a and NIST DB4 databases. An indexing algorithm that uses pores on fingerprint has been proposed by [151]. The features of pores have been extracted by applying Delaunay triangulation. These feature vectors have been clustered using unsupervised k-means algorithm. During retrieval, some clusters are short-listed based on their Euclidean distance with the query fingerprint. The limitation with these approaches is that the global features are not suitable for handling distortions such as occlusion, shear, translation, rotation, scale etc. Also, texture-based techniques require the fingerprint to be aligned prior in the database and that the locations of singular points can be used. However, in low-quality images, it is challenging to locate such points, and hence, these kinds of images get rejected [152].

5.1.2 Minutiae Based Indexing Techniques

This section discusses those indexing techniques that use local features such as minutiae points to construct feature vectors. Each minutia is represented as a triplet containing $\{x_m, y_m, \theta_m\}$ where, the location is represented by *x* and *y* coordinate and θ gives the direction of minutiae in terms of angular value ranging between [0, 2π]. These techniques are further classified into five sub-categories as below.

1. **Single Minutia Based Techniques:**
   These techniques use single minutia for feature vector construction. A minutiae-based geometric hashing has been proposed in [153]. It includes two stages, indexing, and searching which is performed with linear time complexity. This approach builds a fixed-length feature vector from minutia, called as Minutia Binary Pattern. Every minutia and feature vector are inserted into the hash table exactly once, reducing memory and computational cost. The proposed technique has been tested on FVC2004 DB1_A. Single minutia-based techniques are invariant to global distortions such as translation and rotation, and the complexity of the proposed algorithms is also linear.

2. **Minutiae Pair Based Techniques:**
   A lone minutia is not tolerant to elastic deformation. On the other hand, minutiae pair does not suffer from rigid transformations. Exploiting redundant combinations of minutiae pair also provide immunity against noise [154]. A fingerprint indexing technique based on minutiae pair has been proposed in [154]. The features used are the distance between the points, the difference between their angles, and the other two angles formed between their orientation and the line joining them. The

technique has been tested on the FVC2004 DB1_A database. An indexing technique in which features were extracted from minutiae pair has been presented in [155]. The generated feature vector is alignment-free and holds multiple template independence, non-invertibility, and revocability properties. The technique uses densely infinite-to-one mapping (DITOM) in order to achieve these properties. Although both of these techniques are secure, they are not highly accurate. Barman et al. [156] proposed an indexing technique in which the feature vector is generated by computing Euclidean distance between pairs of minutiae points to account for increasing computational complexity. Due to this reason, this technique requires less memory and low computation cost. It was evaluated on FVC2004 DB2_A, DB3_A, and DB4_A databases.

3. **Minutiae Triplet Based Techniques:**

   These techniques consider three minutiae or triangles formed using three minutiae for feature vector construction to index the database. The concept of triangles for fingerprint indexing is given in [157]. The authors considered all possible minutiae triangles, and pairwise matching is done using transformation parameter clustering to increase accuracy. An attempt was made to improve the previous approach by employing new features to minutiae triplets in [158]. It has used handedness, angle, direction, type, maximum side, and geometric constraints based on minutiae characteristics to remove false correspondences. The drawback with these techniques is their high time complexity of O(n^3), where n is the number of minutiae, triangles need to be considered during matching, which makes it computationally expensive.

   A Delaunay Triangles-based technique has been proposed in [159] to reduce the number of minutiae triangles under consideration and hence, save memory. Delaunay triangles are unique and can be computed efficiently. The proposed technique consideres only n minutiae triangles thereby, reducing the time complexity from O(n^3) to O(n). The presence of missing or spurious minutiae in delaunay triangles can lead to the introduction of spurious triangles or missing the important ones [160]. To overcome this problem, a hierarchical indexing technique has been proposed in [160] which divides minutiae into levels based on their quality, and matching is performed starting from the higher level of hierarchy to the lower level. Finally, it accounts for missing and spurious minutiae by combining different enrollment feature sets into a super-template. Another hierarchical indexing technique using barycenter has been proposed by [161] to account for false identification. In this technique, two fingerprints are matched by comparing the angles of triangles extracted from both templates. This ensures that matching is robust to change in orientation. However, the problem here is that some similar triangles may be found that do not belong to the same minutia points. To overcome this problem, the authors have extracted the barycenter of each triangle to ensure that at least three similar triangles are correctly located.

It is known that any slight shift in minutiae points may result in a change of Delaunay triangles, making it sensitive to distortions. A new indexing technique based on features of low-order (order-0 and 1) delaunay (LoD) triangles such as handedness, maximum edge, angles, and related angle between edges and orientation field has been proposed in [162]. In this technique, an extended set of triangles is proposed, containing minutiae points that constitute the LoD. The use of both minutiae and LoD makes it insensitive to elastic distortions. However, the main drawback is that it generates significantly less number of triangles which may be insufficient for fingerprint indexing. Also, some geometric features get lost, making it less accurate [163]. To overcome this limitation, a strategy that is an extension of Delaunay triangles has been proposed in [163]. It uses Delaunay triangles with orders greater than one so that the triangles contain sufficient geometric information. This ensures that even if the number of generated triangles is less in number, the geometric information would be enough to identify a fingerprint. An improvisation of this technique has been presented in [164]. The relative position between core point and all minutiae points is added in the feature vector set. The addition of the new feature resulted in a reduction in the number of erroneous matches and thus, increased accuracy. However, this technique fails to deal with partial fingerprints, in which the core point is missing. A partial fingerprint is shown in Figure5.1(c). The problem with these techniques is that their feature vector is constructed based on ridge counting, which gets incorrect based on the presence of ridge gaps/scars (shown in Figure5.1(b)) at crossing of ridge-counting line and ridge.

4. **Minutiae Quadruplet Based Techniques**

   Minutiae quadruplets get formed by joining four minutiae points. This structure provides features that are robust to distortions as compared to minutiae triplets. An indexing approach based on minutiae quadruplets proposed by [165] uses seven geometric features such as height, the difference of opposite internal angles, diagonals, and area for feature vector construction. k-means clustering is implemented on the feature vector set, and an indexing string is obtained for every image. An indexing string is created for the query image during retrieval, and the table is sorted in decreasing order of the number of quadruplets. All the clusters that contain at least 60% of the quadruplets are considered for matching. This work is extended in [166] by using minutiae quadruplets along with k-means clustering for multi-fingerprint indexing. Indexing techniques based on minutiae quadruplets may be accurate, but they suffer from high computational complexity of $O(n^4)$.

5. **MCC Based Techniques:**

   The minutiae cylinder code is a 3-D data structure that shows the spatial relationship between the distance and orientation of neighboring minutiae. Many techniques consider minutia orientation and other features such as singularities, ridge curvature etc. However, since minutia information is commonly used and is most robust, it is considered favorable to extract features through minutia in the case of large databases [167]. A Locality Sensitive Hashing (LSH) [115], based on minutia cylinder code (MCC) [168], has been proposed in [169] for indexing fingerprint database. Another MCC-based indexing technique has been proposed in [170] that is based on pose estimation. It is assumed that the poses remain invariant among different images of the same fingerprint. Therefore, it is viable to align fingerprints efficiently. The technique has been tested on FVC and NIST databases. Being high dimensional and redundant, MCC does not have a discriminative representation. An indexing technique using a feature mapping matrix to map a real-valued high dimensional version of MCC into a low dimensional binary code has been proposed in [147]. The main objective is to remove redundant information by minimizing inter-bit variation and maximizing intra-bit variation. To reduce the search space, authors have used multi-index hashing, which finds exact k-nearest neighbors on the binary codes.

5.1.3 Hybrid Indexing Techniques

An indexing technique based on both local and global features, ridgeline orientation, frequency aligned across core points has been proposed in [149]. This technique has been tested on FVC2002 DB1 database. An indexing technique that works for partial fingerprint has been proposed in [171]. This technique has been tested on FVC2002 DB1a and DB2a databases. An indexing technique that uses MCC and minutiae vicinity to construct the feature vector has been propounded in [172]. The algorithm then uses support vector machines (SVM) to classify the feature vector into five distinct classes: arch, right loop, left loop, whorl, and tented arch. An indexing technique based on minutiae points and locality sensitivity hashing has been proposed in [173]. After the creation of the feature vector, spectral clustering [174] has been applied, which automatically divides NIST 4 special dataset into 25 classes. For matching, the authors used Euclidean distance, Minkowski distance, and cosine similarity. [175] combined minutiae triplets (Delaunay triangles) and minutiae vicinity to the index fingerprint database. The authors used the direction and location of minutiae points and the density of ridge, and the average value of ridge curvature around minutiae to form a feature vector. The authors also combined this indexing technique with MCC based indexing technique to attain better accuracy and called it score-level fusion.

5.1.4 Deep Neural Network Based Fingerprint Indexing Techniques

A deep convolutional neural network (CNN) is an architecture suitable for filtering an image in a parallel way. They are used in image processing and have contributed significantly to image classification, object detection, image segmentation etc. [176]. To overcome the limitation of minutiae-based indexing techniques, a CNN based indexing approach has been proposed in [177] that generates a fixed-length feature vector without extracting the minutiae points explicitly. It has also used an orientation field dictionary to align the fingerprints into a combined coordinate system. The CNN has been trained using a longitudinal fingerprint database whose last fully connected layer outputs a fixed-length feature vector that forms the index. Due to its fixed length, the feature vector is also useful for template protection. Experimental results over two rolled fingerprint databases, NIST SD4 [178] and NIST SD14 show an error rate of 0.40% and 0.26% at a Penetration Rate of 10%. Another deep neural network-based indexing approach has been proposed in [179], in which real-valued MCC structure is given to the network, and the network outputs a compact binary MCC. The advantage of these binary codes is that they can be directly used as addresses of the hash table, increasing search speed. Then, a multi-index hashing technique is used to speed up the process further.

5.2 Proposed Technique

The proposed technique involves four components. The first component aims to extract core point and minutiae points from a fingerprint image. The second component is feature extraction that involves Coaxial Gaussian Track Code (CGTC) vector construction for every minutia point. The third component is index table generation followed by retrieval of small-sized candidate list in the last component. A block diagram depicting the proposed technique is shown in Figure 5.2. Therefore, the first step in the proposed technique is to extract core point and minutiae points from an input fingerprint. Minutiae points are the points where a ridge ends or bifurcates. These points may vary with sample collection at different points in time. Therefore, to use minutia information for classification and

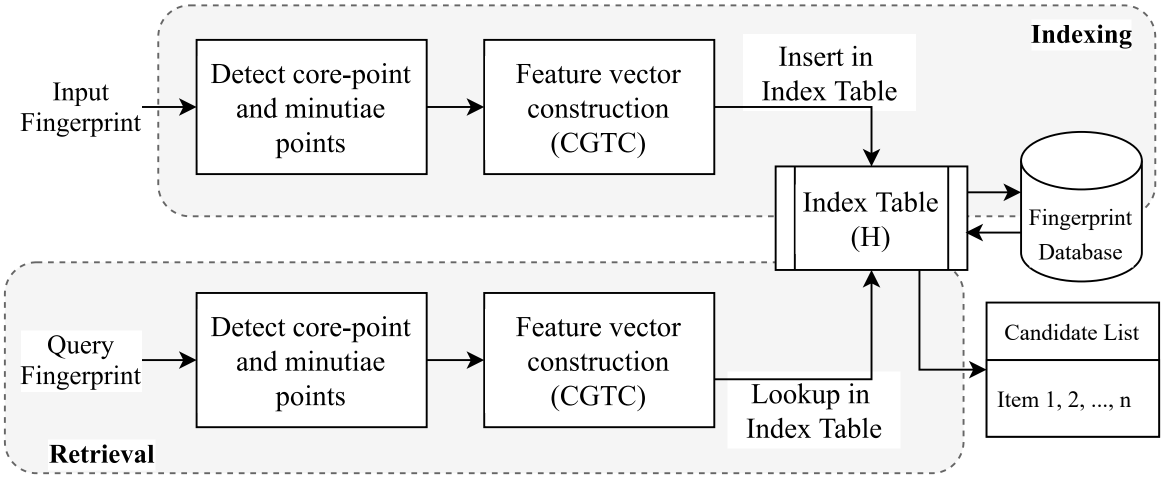


Figure 5.2: Block diagram of the proposed fingerprint indexing technique.

indexing, it is localized with respect to a reference point that remains invariant to rotation and translation. Core point acts as the reference point in a fingerprint [180]. The spatial and directional relationship between the core point and each minutia point is encoded to construct the feature vector. The generated fixed length feature vector is used to generate a quantized look up table. The proposed indexing technique is rotation and translation invariant.

5.2.1 Core-point Detection and Minutiae Extraction

The core point is a particular location in a fingerprint that has high curvature properties. It is a point where the innermost ridge loops are at their steepest [181]. Some fingerprint samples with marked core-point are shown in Figure 5.3. Accurate and efficient detection of the core point plays a significant role in successful feature vector construction. This section proposes a novel deep learning-based architecture for end-to-end core point detection in a fingerprint image. The advantage of the proposed model is that it determines the core point efficiently and in one go.

5.2.1.1 Macro-Localization Network

Macro-Localization Network (MLN) is a U-Net [71] based architecture that maps the input fingerprint image to an image containing the segmented area having the highest probability of containing the core point. The U-Net consists of a contracting and an expansive

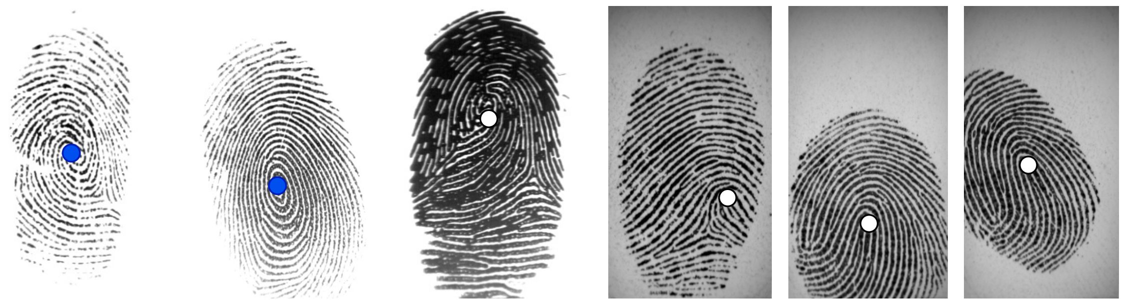


Figure 5.3: Core points shown in fingerprint images taken FVC2004 DB1_A and FVC2002 DB2_A databases (three images each).

path. The contracting path consists of a series of convolutional layers followed by the max-pool layer. On the other hand, the expansive path takes the output of the contracting path and puts it through a series of transposed convolutional and up-sampling layers. The high-level features from the contracting path are concatenated with the corresponding up-sampled features in the decoder using merge connections. U-Net is commonly used in applications involving the segmentation of medical images. When fingerprint images were passed through the standard U-Net for segmenting out the RoI, the observed output is irregular and blurry. This was due to occlusion, pose, and scale variation introduced in the images while acquiring them in an uncontrolled environment. This limitation is addressed in this work by proposing the use of stacked hourglass networks between the contracting and expansive path. Hourglass network [72] includes a residual module that works on the feature vector, unlike the U-Net, which takes a complete image as an input. The residual module uses convolutional operation to learn high-level features, but it is also capable of retaining the original information using skip connections. It has a symmetric topology so that the features are extracted and consolidated across various image scales and resolutions. The output of the hourglass network is of the same size as its input feature vector. Therefore, we can say that the hourglass network only changes the depth of the data without altering its size.

MLN has three parts $viz.$, contracting path (CP), expansive path (EP) and the hourglass network (H). The CP consists of pairs of 3×3 convolutional layer having filter size of 16, 64 and 128 respectively. Every pair of convolutional layer is followed by a

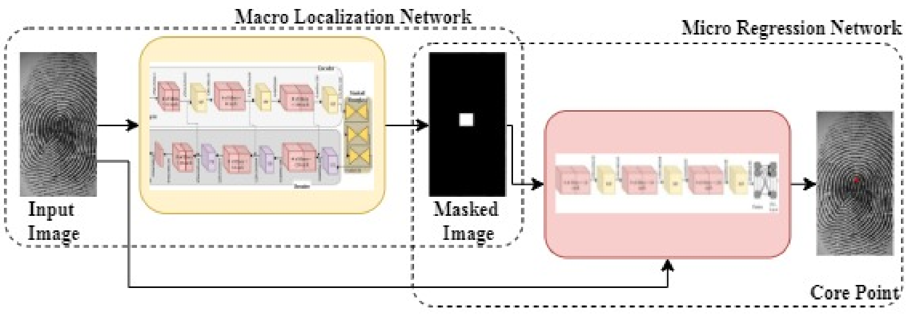


Figure 5.4: Block diagram of the proposed core point detection network.

2×2 max-pool layer. The bottleneck is introduced in the form of three-stacked hourglass networks between the CP and EP. An hourglass network has the ability to capture features at multiple scales and combine them to make pixel-wise predictions. This is made possible by the use of skip-connections that conserve information at every scale. An hourglass network consists of a series of convolution layers followed by a max-pool. This is repeated to process features till a very low scale. The lowest resolution features are then up-sampled. Skip connections are introduced to merge the feature maps at two different scales. The architecture of the single hourglass network is separately shown in Figure 5.6. Adding hourglass modules subsequently one after the other allows for re-assessment of the features across the whole image. It also allows for going to-and-fro between the scales which further helps in conserving spatial relationship among features. Directly passing the encoded image to the EP may result in an inappropriate localization, especially in the noisy images. Adding a stacked hourglass network makes the network robust to such situations. MLN was first trained with only one hourglass network as a bottleneck. The network showed improvement in result when two and then three hourglass networks were added. However, there was not much improvement when fourth hourglass network was stacked in the bottleneck. Hence, MLN has three stacked hourglass networks as the bottleneck. The output of the bottleneck is passed to the EP that performs segmentation on the input image. The EP network first performs up-sampling using a 3×3 transposed convolution layer. Its output is concatenated with the corresponding feature map of the CP using the merge connections. The merge connections preserve the essential

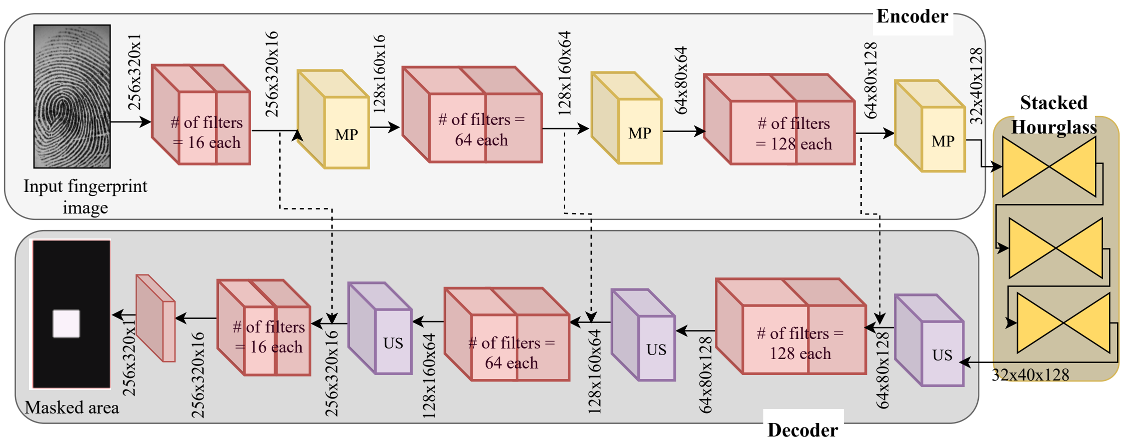


Figure 5.5: Architecture of Proposed Macro Localization Network

spatial information of the input image that may have got lost in the encoder. Using merge connections also eliminates the problem of vanishing gradient during learning. The concatenated feature map is passed to a pair of 3×3 convolution layer followed by ReLU activation. This is repeated for different filters of size 128, 64 and 16. The complete architecture of the network is shown in Figure 5.5.

5.2.1.2 Micro Regression Network

The output of MLN is the segmented image that has a white region depicting the probable region that contains the core point. Micro Regression Network (MRN) takes the original fingerprint image and the output of the MLN concatenated along channel dimension as the input and outputs two values depicting the coordinates of the core point. The original image is passed because the output of MLN is a white area and the corresponding fingerprint patch is required to further trace down the location of the core point in that region. MLN consists of three convolutional blocks of 3×3 filters with varying number of filters $i.e.$ 16, 64 and 128. Each block is followed by ReLU activation and a 2×2 max-pool layer. The feature map of the last maxpool layer is flattened and fed through four fully connected layers. The last fully connected layer outputs two values corresponding to the predicted (x, y) coordinates of the core point. This network mainly performs regression on the proposed segmented region and outputs the location of the core point in a given fingerprint image. The architecture of the Micro Regression network is graphically shown

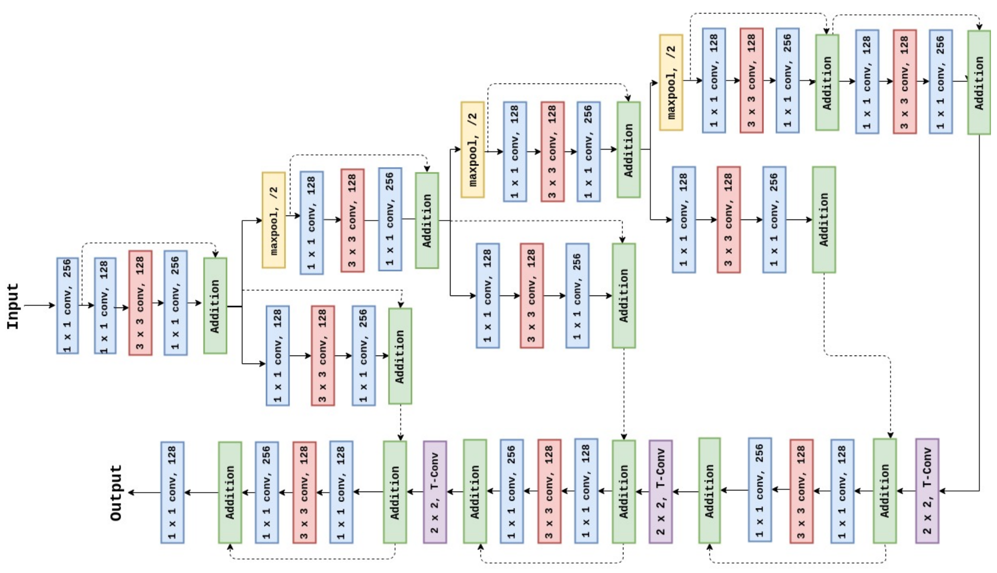


Figure 5.6: Hourglass Network Used in Proposed Model (Dotted lines represents skip connections)

in Figure 5.7.

All these network components are stacked one over the other after training them individually to build a single network that predicts the location of the singular point in a given fingerprint image. Binary cross entropy and mean squared error are used to train the proposed network. The cross entropy loss is back-propagated by the macro-localization network to learn the proposed region for singular point presence while the mean squared error is back-propagated to the regression network for learning core point localization.

5.2.1.3 Network Training

Training of this network is broadly divided into two steps. First, the MLN is trained that takes a fingerprint image as input and produces a segmented mask containing the core point of the fingerprint. This network is trained using cross entropy loss function that computes the pixel-wise difference between the ground truth mask and the predicted mask. Finally, the computed loss is averaged over all the pixels in the image. Let, for a given fingerprint image Y of size $h \times w$, the ground truth and predicted mask is denoted by Y_m and $\hat{Y}_m$ respectively. The loss between Y_m and $\hat{Y}_m$ is defined as given in the equation below.

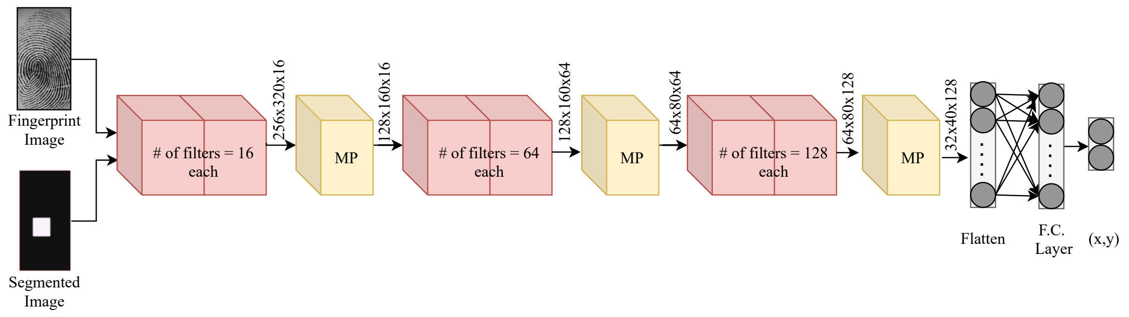


Figure 5.7: Architecture of the Proposed Micro Regression Network

$$L_{CE}(\hat{Y}_m, Y_m) = \frac{-1}{h * w} \sum_{j=1}^{h} \sum_{i=1}^{w} \Big[Y_m(i,j) * \log(\hat{Y}_m(i,j)) + (1 - Y_m(i,j)) * \log(1 - \hat{Y}_m(i,j)) \Big] \tag{5.1}$$

In the second step, the MRN is trained separately by minimizing the mean squared error (MSE) between the actual coordinates of the core point and the one predicted by the proposed network. The loss function in mathematically expressed in Eq.(5.2). Here, $Y_{(x,y)}$ and $\hat{Y}_{(x,y)}$ denote the actual and predicted (x, y) coordinates of the actual and predicted core point. The loss is averaged for all the fingerprint images, denoted by N. After training both the networks separately, they are combined by stacking to form a single end-to-end core point detection network. This network is then trained in an end-to-end manner.

$$L_{MSE}(Y_{(x,y)}, \hat{Y}_{(x,y)}) = \frac{1}{N} \sum_{i=1}^{N} ||Y_{(x,y)} - \hat{Y}_{(x,y)}||^2 \tag{5.2}$$

5.2.2 Feature Extraction and Indexing

Coaxial Gaussian Track Code (CGTC) is a fixed length feature vector that is constructed using the location and angle information of a minutia point contained in a fingerprint. Given a fingerprint template M that has n minutiae points, $\{m_1, m_2, \ldots, m_n\}$; the CGTC vector of length p is constructed for a given minutia point m_i by accommodating binary

Algorithm 7 CGTC (M, i)

```
Require: minutiae template $M$
Ensure: $p$ bit CGTC vector for a minutia point $m_i \in M$
1: for $t = 1$ to $p$ do
2:     $m_{i_{(CGTC)}}[t]$ = 0
3: for $t \in \{1, 2, ..., i-1, i+1, ..., n\}$ do
4:     $d = \sqrt{(m_i.x - m_t.x)^2 + (m_i.y - m_t.y)^2}$
5:     for $j = d - a$ to $d + a$ do
6:         $m_{i_{(CGTC)}}[j]$ += $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-b(\frac{j-d}{\sigma})^2}$
   return $m_{i_{(CGTC)}}$
```

information of minutiae points in its p-pixels neighborhood. The process starts off by geometrically rotating the fingerprint template such that the orientation of core point becomes parallel to the horizontal or $x-$axis. The fingerprint is then partitioned into 72 sectors around the core point after every 5°. The spatial and directional information about a minutia point m_i in M is obtainable by figuring out the sector in which m_i lies and its Euclidean distance from the core point. Each minutia point is now uniquely identifiable by the sector number and distance, denoted by $m_i.s$ and $m_i.d$ respectively.

A p-bit CGTC vector for each minutiae m_i in M is constructed by computing Euclidean distance between m_i and every other minutia point, say m_t, where $t = \{1, 2, \ldots, n\}$. Gaussian is applied on the computed distance lying in the 'a' neighborhood pixels. The process is also described in Algorithm 7. The constructed CGTC vector for m_i minutia is inserted into the lookup table at location ($m_i.s$, $m_i.d$). This technique is rotation and translation invariant as the minutia point can be uniquely identified with respect to the core point. This process is described in Algorithm 8.

5.2.3 Retrieval

During retrieval, the minutiae points and the core point are extracted from the query fingerprint, denoted by Q. Let M_Q denotes the set of k minutiae points extracted from the query fingerprint, where $M_Q = \{m_1, m_2, \ldots, m_k\}$. The query fingerprint is geometrically rotated to align it parallel to the horizontal axis. The CGTC vector for the query fingerprint is constructed using the Algorithm 7. Then we compute the sector and dis-

Algorithm 8 Indexing (H, M)

```
Require: A 2D lookup table $H$ and a fingerprint image $M$
Ensure: Updated $H$
1: Extract minutiae points from $M$
2: Detect the core point $C = (c.x, c.y, c.\theta)$ of $M$
3: Rotate the fingerprint template to make the core point parallel to x-axis
4: Geometrically transform the minutiae points
5: Divide $M$ into 72 sectors
6: for minutiae point $m_i$, $i = \{1, 2, \ldots, n\}$ do
7:     Construct CGTC vector of $p$ length
8:     Determine sector $m_i.s$ and distance $m_i.d$
9:     Insert $ID(M)$ and $m_{i_{(CGTC)}}$ at $(m_i.s, m_i.d)$ into $H$
   return $H$
```

tance of each minutia point lying in M_Q with respect to the core point. Let $m_k.s$ and $m_k.d$ be the sector and distance respectively of a minutia point m_k. The index table is referred to at location $(m_k.s, m_k.d)$ and the fingerprint IDs and CGTC vectors of the minutiae points lying in that location are retrieved for comparison. The query fingerprint may have translated, missed, spurious or rotated minutiae points. Therefore, along with the selected bin in the index table, $i.e.(m_k.s, m_k.d)$, neighboring bins $\delta s \times \delta d$ are also considered for similarity computation and candidate set generation. The similarity is computed between m_k's CGTC vector and the CGTC vectors of all the retrieved candidate fingerprint IDs. A separate score list is generated for each minutia point in Q. Hence, this process results in k score lists, one for each minutia point. Finally, we select only those fingerprint IDs that have similarity score greater than a specific threshold value. All the k lists are concatenated to form a single score list S for a fingerprint query image Q $i.e.$, $S = \{S_1, S_2, \ldots, S_k\}$. Top t fingerprint IDs with respect to the score in S are selected for identification. The process is also explained through Algorithm 9.

5.3 Experimental Results

This section discusses the specifications of the fingerprint databases used followed by a brief overview of the evaluation parameter utilized for evaluating the performance of the proposed core point detection model. The proposed technique starts by extracting

Algorithm 9 Retrieval (H, Q)

```
Require: A 2D lookup table $H$ and a query fingerprint image $Q$
Ensure: Candidate list for $Q$
 1: Extract minutiae points
 2: Detect the core point
 3: Rotate the fingerprint template to make the core point parallel to x-axis
 4: Geometrically transform the minutiae points
 5: Divide $Q$ into 72 sectors
 6: for minutiae point $m_k$, $k = \{1, 2, \ldots, m\}$ do
 7:     Construct CGTC vector of $p$ length
 8:     Determine sector $m_k.s$ and distance $m_k.d$
 9:     $S_{m_k} = \phi$
10:     for all $m_j \in \delta d \times \delta s$ neighbours of $(m_k.s, m_k.d)$ do
11:         Extract CGTC vector $m_{jCGTC}$ of $m_j$ from $H$
12:         if $dist(m_k, m_j) < Th$ then
13:             $S_{m_k}[ID(Q)^j].score = S_{m_k}[ID(Q)^j].score \times w$
14:     for all $m_j \in \delta d \times \delta s$ neighbors of $(m_k.s, m_k.d)$ do
15:         $S_{m_k}[ID(Q)].score$ += $S_{m_k}[ID(Q)^j].score$
16: Arrange $ID(Q)$s in decreasing order of score, for each minutia point $m_k$ in $Q$
17: Concatenate $S_{m_k}$'s for $k$ minutiae points in $Q$
    return Candidate List
```

core and minutia points from the fingerprint images. Minutiae points are extracted using a standard library but the location of the core point is detected by using the proposed model. Therefore, this section states the obtained experimental results for the 1) proposed core-point detection model and 2) proposed indexing algorithm on two publicly available fingerprint databases $viz.$ FVC2002 DB2_A [182] and FVC2004 DB1_A [183].

5.3.1 Database

Two publicly available fingerprint databases are used to evaluate the proposed technique. The databases were originally collected and used to host fingerprint verification competition. The specifications of the databases are given as follows.

FVC2002 DB2_A [182]: This database contains 800 fingerprint images collected from 100 subjects. Eight fingerprint impressions of the same finger were collected from each individual using FX2000 sensor by Biometrika. The images have a size of 296×560 pixels with a resolution of 560 dpi. The images in this database have a huge intra-class

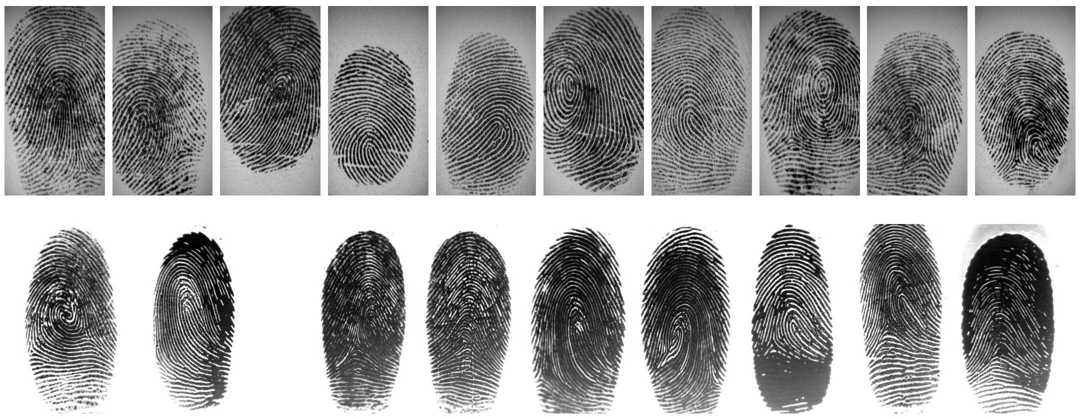


Figure 5.8: Sample fingerprint images from FVC2002 DB2_A (first row) and FVC2004 DB1_A database (second row).

variation among different impressions of the same subject due to external factors such as placement of fingerprint, pressure etc. Some sample images are shown in the first row of Figure 5.8

FVC2004 DB1_A [183]: This is a publicly available database. It consists of 800 fingerprint images collected from 100 subjects using "U.are.U 4000" sensor by Digital Persona. The images have a size of 640×480 pixels and a resolution of 500 dpi. Some of the fingerprint images from FVC2004 DB1_A database are shown in the second row of Figure 5.8.

5.3.2 Experimental Setting

The implementation of the proposed network has been done on a Linux based operating system with NVIDIA GeForce GTX 1080 Ti graphics card with graphics memory of 11 GB. Adam optimizer with a learning rate of 0.0005 has been used to train the core point detection model. The network is trained for 100 epochs with a mini-batch of 8 images, each of size 256×320. The parameters considered for the indexing technique are CGTC vector length of 50, neighborhood bin value $\delta d = 6$ and $\delta s = 3$ and lastly the radius for CGTC vector is 3.

5.3.3 Results

This section shows the results obtained for core point detection and indexing of fingerprint databases using the proposed technique. True Detection Rate (TDR) is used for evaluating the performance of the core point detection model. The indexing technique is evaluated in terms of Hit Rate and Penetration Rate which are described in Chapter 1. Both the results are discussed separately in the following sub-sections.

5.3.3.1 Core Point Detection

True Detection Rate (TDR) is used as a measurement index to gauge the performance of the proposed technique. The trained model outputs (x, y) coordinates of the core point in the fingerprint sample. The coordinates will be considered accurate if the euclidean distance between the original and predicted coordinate is less than or equal to 20 pixels. This can be formulated as given in the equation below. Here, ($\hat{C}.x$,$\hat{C}.y$) and ($C.x$,$C.y$) refer to the (x, y) coordinates of the ground truth and the predicted core point respectively.

$$\sqrt{[C.x - \hat{C}.x]^2 + [C.y - \hat{C}.y]^2} \leq 20\, pixels \tag{5.3}$$

The proposed model is trained on 80% of both the datasets and tested on the remaining 20%. The core point is manually annotated in all the images for preparing ground truth for training the proposed model. The first result shows the values of TDR achieved when the Euclidean distance between predicted and ground truth coordinate is 20 pixels. The proposed technique achieved true detection rate (TDR) of 98.75% for FVC2002 DB2_A database. The proposed technique shows a detection rate of 96.25% on FVC2002 DB2_A database when the difference in distance between the ground truth and the predicted coordinates is not more then 10 pixels.

The obtained values are compared with other state-of-the-art techniques. Zhou et al. [184] proposed a singular point detection technique for deltas and core points. The technique uses Differences of the ORIentation values along a Circle (DORIC) features which is an extension of Poincare index [185]. Xie et al. [186] proposed an inconsistency feature to detect core point from fingerprint images. The feature is obtained by deter-

Table 5.1: Comparison of core point detection results on FVC2002 DB2_A database with state-of-the-art techniques

Technique	Description	TDR
Zhou et al. [190]	Orientation values along a circle (DORIC)	95.78%
Xie et al. [186]	Inconsistency feature	90.00%
Tiwari et al.[187]	Meandering energy potential (MEP)	95.75%
Liu et al. [188]	Faster R-CNN	96.03%
Proposed	MLN and MRN	96.25%

mining the relationship between ridge lines and curves present in a fingerprint sample. The proposed technique enhances the fingerprint images using a method that combines Short Time Fourier Transform (STFT) Analysis with a quality estimation method. It is followed by core point detection in the enhanced images using the posterior probability. An approach utilizing Meandering Energy Potential (MEP) for detection of core point has been proposed in [187]. MEP refers to the amount of shield at a particular pixel in the fingerprint image. The advantage of this approach is that it does not require any prior knowledge of the fingerprint structure under evaluation. A deep learning based core point detection approach based on Faster-RCNN has been proposed in [188]. Faster-RCNN [189] generates region proposals that potentially contain a singular point. The approach then considered top-100 region proposals for singular point detection. Table 5.1 presents the comparison of TDR of the proposed technique with other techniques proposed in literature taking 10 pixels in consideration. It is clearly evident that the proposed technique outperforms all other techniques.

5.3.3.2 Indexing

Indexing performance is evaluated in terms of hit rate and penetration rate. It is expected from a good indexing technique to achieve lower penetration rate at high hit rate. The penetration rate is computed for the proposed technique at various values of hit rate. Figure 5.9 is a graph showing the relationship between hit rate and penetration rate. The proposed technique achieves a penetration rate of 0.79% for 99% hit rate. While, only 0.86% of the database is required to find the true match of the query fingerprint sample with 100% confidence.

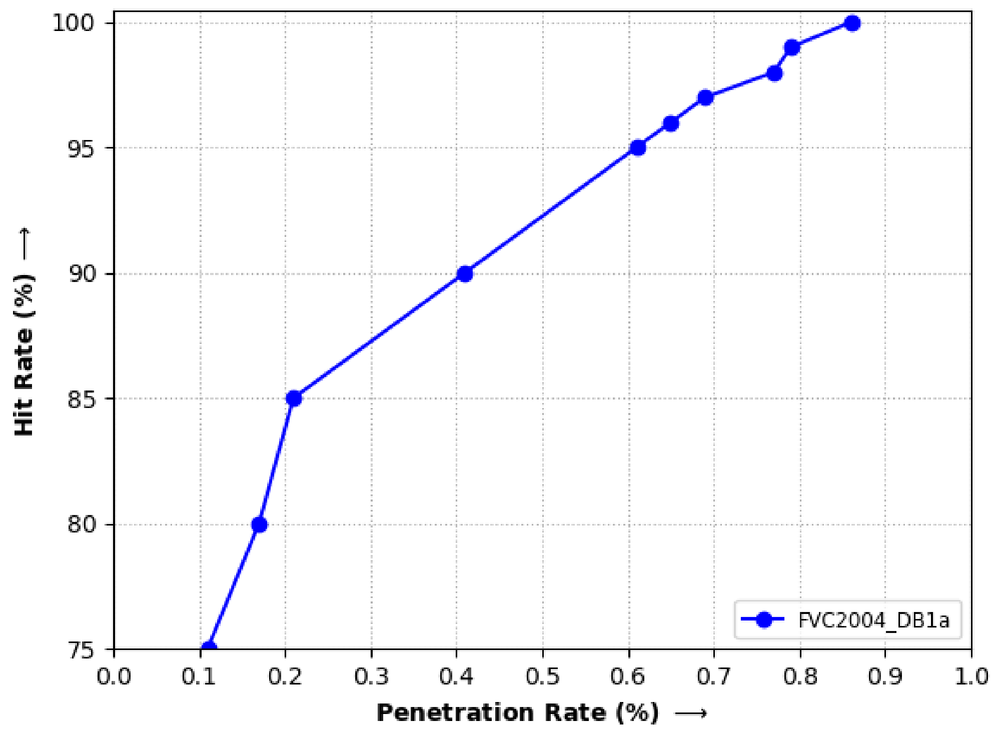


Figure 5.9: Graph showing relationship between hit rate and penetration rate obtained for FVC2004 DB1_A database using the proposed technique.

The proposed technique is compared with other state-of-the-art techniques to demonstrate its efficiency. The comparison is made with respect to two parameters. First one being the hit rate and penetration rate. Table 5.2 compares the penetration rate achieved by various fingerprint indexing techniques on different values of hit rate. The techniques considered in table are minutiae based techniques. Bhanu et al. [191] proposed a minutiae triplet based technique. On the other hand, Bebis et al. [159] and Liand et al. [192] used Delaunay and Low order Delaunay triangles respectively for indexing fingerprint database. These two techniques performed better in terms of low penetration rate. Lastly, Iloanusi et al. used minutiae quadruplets and achieved an even lower penetration rate. However, it is clearly evident that the proposed technique achieves the lowest penetration rate at all the three values of hit rate. The proposed technique requires only 0.86% of the FVC2004 DB1_A database to identify a query fingerprint with 100% confidence.

Table 5.2: Comparison of the proposed technique with state-of-the-art techniques in terms of Penetration Rate obtained at various values of Hit Rate.

Database	*Penetration Rate* at *Hit Rate* (HR)		
FVC2004 DB1_A	HR = 95%	HR = 99%	HR = 100%
Minutiae Triplets [191]	8.1 %	27.2 %	40.09 %
Delaunay [159]	7.2 %	18.1 %	32.7 %
LoD Triangles [162]	3.6 %	10.0 %	20.9 %
Quadruplets [165]	-	11.8 %	12.0 %
Proposed method	**0.61 %**	**0.79 %**	**0.86 %**

Table 5.3: Comparison in terms of Average Penetration Rate (APR) of the proposed technique with other state-of-the-art techniques

Author	**Technique**	**APR (%)**
Bai et al. [147]	Binary Fingerprint Descriptor	4.83
Bai et al. [179]	Deep Compact Binary Minutia Cylinder Code	4.58
Kavati et al. [193]	Hierarchical Decomposition of Extended Triangulation	9.70
Lee et al. [194]	Extended Triangulation	4.31
Proposed	**CGTC**	**0.56**

Another comparison of the proposed technique with the state-of-the-art techniques is done in terms of average penetration rate (APR). APR is computed by stopping the search as soon as the true match of the query fingerprint is found out and determining its rank. Table 5.3 compares the proposed technique on the basis of APR (%). Bai et al. proposed two techniques in [147] and [179] that overcome the limitation of MCC by converting the real-valued MCC into a low-dimensional binary code. The technique proposed in [147] achieves an APR of 4.82% while the other technique [179] that proposed a deep learning based binary minutiae cylinder code (DCBMCC) achieved a lower APR of 4.58%. Lee et al. proposed an improvisation to the technique proposed in [194] by presenting a new index vector that has un-correlated elements so that the index values are uniformly distributed over the search space. This reduces the number of comparisons by retrieving only a small list of candidates. The technique achieved an APR of 4.305% which is better than the technique proposed in [193]. It can be seen from the results shown in Table 5.2

and Table 5.3 that the proposed technique achieves lowest penetration rate and APR and therefore, is the most efficient as compared to the techniques proposed in the literature.

5.4 Summary

This chapter proposes an efficient technique to index a fingerprint database by learning a feature vector that makes use of directional and spatial information between the core point and minutiae points. The first component aims at detecting the location of the core point for which a novel autoencoder-based architecture with a stacked hourglass has been proposed. Later, the fingerprint is divided into 72 sectors of five degrees each. The sector number of each minutia point along with its distance from the core point is recorded to construct its feature vector. The constructed feature vectors are stored in (x,y) location of the index table, where x denote the sector in which the minutia point lies and y represents its distance from the core point. During retrieval, the same procedure is followed for the query fingerprint. The location of each minutia point in the query fingerprint is decoded and all the feature vectors lying in that location are retrieved for similarity comparison. This is repeated for each minutiae point and a similarity list is maintained showing the most similar ones at the top. A cumulative similarity list is generated by combining all the lists generated for the minutiae points to retrieve top-k candidates for identification. The proposed technique is tested on FVC2004 DB1_A database and was found to achieve an Average Penetration Rate (APR) of 0.56$\%$ which is better than other techniques proposed in the literature.

Chapter 6

Conclusions and Future Directions

6.1 Conclusion

This thesis addresses the problem of accelerating the identification process in biometric databases. It is vital to get identification responses in real-time, especially for large databases. The objective of identification is to establish the identity of a query sample by comparing it with all the templates stored in the database by finding the most similar one. The presented indexing techniques output a candidate list for comparison against the given query biometric sample. The generated candidate list is small and fixed in size with respect to the database under consideration. This results in making the identification process a constant time operation. We have considered biometric databases of four biometric modalities $viz.$ iris, palmprint, finger knuckleprint, and fingerprint. Deep learning-based approaches have been proposed for feature extraction and to generate hash codes for biometric samples. The network architectures and loss functions used in the proposed techniques are customized according to the modality. The results obtained in each chapter for the considered databases are summarized in Table 6.1. The value of the penetration rate with respect to various values of hit rate have been reported for the considered benchmark databases. However, the results can vary (improve or worsen) for different databases. In view of the wide variety of possible application scenarios and databases, no theoretical guarantees should be drawn. The efficiency of the proposed techniques can be derived from the comparison made with the state-of-the-art indexing techniques proposed for the corresponding modality. The conclusion inferred from each chapter, followed by the future scope of this research work, is discussed in the following paragraphs.

The first indexing technique is proposed for iris databases that uses a specialized convolutional neural network architecture trained as a siamese architecture to output a compact feature vector for iris images. The network is trained so that the learned feature vectors have low inter-class and high intra-class similarity in the latent representation space. The proposed technique is an end-to-end pipeline that aims at learning feature vectors to index the iris database for faster identification. k-means and agglomerative clustering are explored to generate an index table. During retrieval, the candidates lying in the most similar index as the query image are fetched for comparison. The proposed technique has been tested on CASIA Interval [47] and CASIA Lamp [79] database and achieved a 99% hit rate at just 2.254% and 0.008% penetration rate, respectively. In other words, one needs to search only 2.254% and 0.008% of the considered datasets to be 99% sure about finding the true match of the query sample. A speedup of ~ 4 and ~ 27 times is achieved when the CASIA Interval and CASIA Lamp are indexed using the proposed method compared to the naive approach for identification. A comparison of the proposed technique with the state-of-the-art techniques is made in Table 2.8 which indicates the effectiveness of the proposed technique.

The next technique, called PalmHashNet, presents a palmprint database indexing technique to learn discriminative embeddings for palmprint images. The generated embeddings have high intra-class and low inter-class similarity and are indexed using k-means clustering and Locality Sensitive Hashing (LSH) technique to create an index table. Whenever a query image is given to the identification system, its features are extracted using the trained PalmHashNet. The generated feature vector is compared with all the indices of the index table, and the candidates lying in the most similar index are retrieved for comparison. Identification experiments are conducted on four publicly available popular palmprint databases *viz.* CASIA, IITD-Touchless, Tongji-Contactless and PolyU II palmprint databases. PalmHashNet achieved a penetration rate of 0.022%, 1.032%, 4.555% and 0.39% at 100% hit rate on the considered databases respectively. This implies that we need to only look out for less than 1% of the CASIA Palmprint Image Database [49] and Hong Kong Polytechnic University Palmprint II Database (PolyU II) [52] while 1.03% and 4.55% of the IIT Delhi Touchless Palmprint Database [50] and Tongji Contactless Palmprint Dataset [51] to find the true match of a query sample with 100% confidence. Hence, using PalmHashNet, we need to search only a small percentage of the database instead of the whole database for identification without compromising the recognition accuracy. The proposed technique outperforms other state-of-the-art recognition as well as indexing techniques given in the literature. The proposed technique can create an efficient fixed-size candidate list for comparison, thereby making identification a constant time operation.

An indexing technique called FKPIndexNet that learns discriminative embeddings for finger-knuckleprint images is proposed in Chapter 4. The generated embeddings have high intra-class and low inter-class similarity. Three different techniques *viz.* k-means Clustering, BallTree hashing and LSH have been explored for index table generation. Whenever a query image is shown to the system, the features are extracted using the same method for query FKP image as well. The generated feature vector is matched with all the indices of the index table and the candidates lying in the most similar index are retrieved for comparison. Experiments are performed for verification and identification of the proposed technique on two publicly available benchmark FKP databases *viz.* PolyU-FKP and IITD FKP. A thorough evaluation of the extracted features is presented and analysed in a comprehensive manner. Proposed technique has achieved 100% of hit rate at the penetration rate of only 3.42% and 0.32% for PolyU-FKP and IITD FKP database respectively. Hence, by using FKPIndexNet, we need to search only a small percentage of the database instead of whole database for identification and that too without compromising on the accuracy. Identification using indexed database also provides a speedup of ~ 11 and ~ 4 times on PolyU-FKP and IITD FKP database respectively.

Table 6.1: Summary of the achieved penetration rate (%) at different hit rates after indexing the considered databases using the proposed techniques.

Trait	Feature Extractor	Database(s)	PR at HR = 95%	PR at HR = 100%
Iris	IrisIndexNet	CASIA Interval [47]	0.052%	2.254%
		CASIA Lamp [79]	0.008%	0.008%
Palmprint	PalmHashNet	CASIA Palmprint [49]	0.022%	0.022%
		IITD Touchless [50]	0.054%	0.162%
		Tongji Contactless [51]	0.016%	0.87%
		PolyU II [87]	0.025%	0.025%
FKP	FKPIndexNet	PolyU-FKP [53]	0.02%	0.32%
		IITD FKP [54]	0.02%	0.32%
Fingerprint	CGTC	FVC2004 DB1_A [183]	0.61%	0.86%

Chapter 5 presents a novel indexing technique for fingerprint databases. A feature vector representing a given fingerprint is constructed by encoding the relationship between the core point and each minutiae point. Hence, the first and foremost step is to extract the locations of core and minutiae points from every fingerprint. A novel end-to-end U-Net based network with three stacked hourglass networks has been proposed to detect the core point from a fingerprint image. The model takes a fingerprint image as an input and outputs the core point's location coordinates (x,y). The proposed model consists of two components, a Macro-Localization Network and a Micro-Regression Network. The Macro-Localization Network (MLN) applies segmentation over the entire fingerprint image and learns the probable area containing the core point. The second network, called Micro-Regression Network (MRN), takes the output of MLN along with the original input fingerprint image to regress the segmented area to localize the coordinates of the core point. The minutiae points are extracted using a minutiae detector, called MINDTCT, contained in NIST Biometric Image Software. The proposed work constructs a softmax-based Coaxial Gaussian Track Code (CGTC) technique for fingerprint database indexing. The CGTC vector is constructed for each minutiae point in a fingerprint and these are added to the index table exactly once, making its time complexity linear. The retrieval algorithm returns a small, fixed-size candidate list for comparison with the query fingerprint. This makes the search time a constant-time operation. The proposed approach has been tested on FVC2004 DB1_A [183], and it achieves a 100% hit rate with 0.86% penetration rate.

6.2 Future Work

1. **Acquisition of Large Databases:** This research work aims at designing indexing techniques for biometric databases to facilitate faster identification. However, the benchmark databases that are publicly available and are widely used for experimentation are smaller in size as compared to the actual biometric databases in real world scenarios. Therefore, there is a need to collect a large database for experimentation.

2. **Multimodal Systems:** Biometric authentication systems can be classified as Unimodal or Multimodal based on the number of modalities considered. Unimodal systems involve the use of only one biometric trait while multimodal systems use two or more biometric traits together for authentication. However, to combine different biometric traits into one system require fusion which can be done at sensor, feature, matching score or decision level. In this work, indexing techniques are proposed for unimodal systems. The work can be extended by combining multiple modalities for indexing and identification.

3. **Presentation Attack Detection:** Biometric authentication systems are susceptible to external attacks. The attacks that happen at sensor level by showing an artifact of a biometric sample, such as wooden glue fingerprint or a photo of palmprint or iris or finger dorsal, are known as presentation attacks. With an increase in availability of images and videos online, it is not difficult for the attacker to acquire an individual's personal data. Moreover, no extra knowledge of software or internal working of the system is required to break the authentication system as the attack happens at the hardware level. Therefore, presentation attack detection can be addressed as an extension to this work.

4. **Template Security:** With increase in adaptation of biometric authentication system in almost every device, it has become vulnerable to attacks. The attack can be made on the biometric database. One possibility is to overwrite the original template with an illegitimate sample to get unauthorized access. Other could be stealing

a template from the database and later, spoofing it to be shown to the sensor or using the stolen template to get access to some other device or functionality. The traditional modes of authentication such as passwords or PINs can be cancelled or revoked when they are compromised but biometric templates can't be cancelled. Therefore, there is a need of biometric template security. Various methods are used for template protection but they are mainly classified into feature transformation methods and biometric cryptosystems [195]. Template security is beyond the scope of this work and can be pursued as an extension of the same.

Bibliography

[1] Ruud M Bolle, Jonathan H Connell, Sharath Pankanti, Nalini K Ratha, and Andrew W Senior. *Guide to biometrics*. Springer Science & Business Media, 2013.

[2] Anil K Jain, Patrick Flynn, and Arun A Ross. *Handbook of biometrics*. Springer Science & Business Media, 2007.

[3] Davide Maltoni, Dario Maio, Anil K. Jain, and Salil Prabhakar. Handbook of fingerprint recognition. In *Springer Professional Computing*, 2003.

[4] Jin Bo, Tang Hua Ping, and Xu Ming Lan. Fingerprint singular point detection algorithm by poincaré index. *WSEAS Transactions on Systems*, 7(12):1453–1462, 2008.

[5] Xiangqian Wu, David Zhang, and Kuanquan Wang. Fisherpalms based palmprint recognition. *Pattern recognition letters*, 24(15):2829–2838, 2003.

[6] Somnath Dey and Debasis Samanta. Iris data indexing method using gabor energy features. *IEEE Transactions on Information Forensics and Security*, 7(4):1192–1203, 2012.

[7] Dayong Wang, Charles Otto, and Anil K Jain. Face search at scale. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1122–1136, 2016.

[8] Lin Zhang, Lei Zhang, and David Zhang. Finger-knuckle-print: a new biometric identifier. In *IEEE International Conference on Image Processing (ICIP)*, pages 1981–1984. IEEE, 2009.

[9] Kateřina Přihodová and Miloslav Hub. Biometric privacy through hand geometry-a survey. In *International Conference on Information and Digital Technologies (IDT)*, pages 395–401. IEEE, 2019.

[10] Bogdan Belean, Mihaela Streza, Septimiu Crisan, and Simina Emerich. Dorsal hand vein pattern analysis and neural networks for biometric authentication. *Studies in Informatics and Control*, 26(3):305–314, 2017.

[11] Marcos Faundez-Zanuy. Signature recognition state-of-the-art. *IEEE aerospace and electronic systems magazine*, 20(7):28–32, 2005.

[12] Tyler K Perrachione, Stephanie N Del Tufo, and John DE Gabrieli. Human voice recognition depends on language ability. *Science*, 333(6042):595–595, 2011.

[13] Khalid Bashir, Tao Xiang, and Shaogang Gong. Gait recognition without subject cooperation. *Pattern Recognition Letters*, 31(13):2052–2060, 2010.

[14] Ahmed Awad E Ahmed and Traore Issa. A new biometric technology based on mouse dynamics. *IEEE Transactions on dependable and secure computing*, 4(3): 165–179, 2007.

[15] Monika Bhatnagar, Raina K Jain, and Nilam S Khairnar. A survey on behavioral biometric techniques: Mouse vs keyboard dynamics. *International Journal of Computer Applications*, 975:8887, 2013.

[16] Roy A Maxion and Kevin S Killourhy. Keystroke biometrics with number-pad input. In *International Conference on Dependable Systems & Networks (DSN)*, pages 201–210, 2010.

[17] Phalguni Gupta, Kamlesh Tiwari, and Geetika Arora. Fingerprint indexing schemes–a survey. *Neurocomputing*, 2018.

[18] WF Leung, SH Leung, WH Lau, and Andrew Luk. Fingerprint recognition using

neural network. In *Neural Networks for Signal Processing*, pages 226–235. IEEE, 1991.

[19] John G Daugman. High confidence visual recognition of persons by a test of statistical independence. *IEEE transactions on pattern analysis and machine intelligence*, 15(11):1148–1161, 1993.

[20] Ranjeet Ranjan Jha, Gaurav Jaswal, Divij Gupta, Shreshth Saini, and Aditya Nigam. Pixisegnet: Pixel-level iris segmentation network using convolutional encoder–decoder with stacked hourglass bottleneck. *IET Biometrics*, 9(1):11–24, 2019.

[21] Naser Damer, Philipp Terhörst, Andreas Braun, and Arjan Kuijper. Efficient, accurate, and rotation-invariant iris code. *IEEE Signal Processing Letters*, 24(8): 1233–1237, 2017.

[22] Adams Kong, David Zhang, and Mohamed Kamel. A survey of palmprint recognition. *pattern recognition*, 42(7):1408–1418, 2009.

[23] Aditya Nigam, Kamlesh Tiwari, and Phalguni Gupta. Multiple texture information fusion for finger-knuckle-print authentication system. *Neurocomputing*, 188:190–205, 2016.

[24] Maria De Marsico and Alessio Mecca. A survey on gait recognition via wearable sensors. *ACM Computing Surveys (CSUR)*, 52(4):1–39, 2019.

[25] Piergiorgio Vittori. Ultimate password: is voice the best biometric to beat hackers? *Biometric Technology Today*, 2019(9):8–10, 2019.

[26] Chao Shen, Zhongmin Cai, Xiaohong Guan, Youtian Du, and Roy A Maxion. User authentication through mouse dynamics. *IEEE Transactions on Information Forensics and Security*, 8(1):16–30, 2012.

[27] Jun Yu, Kejia Sun, Fei Gao, and Suguo Zhu. Face biometric quality assessment via light cnn. *Pattern Recognition Letters*, 107:25–32, 2018.

[28] Yi Chen, Sarat C Dass, and Anil K Jain. Localized iris image quality using 2-d wavelets. In *International conference on biometrics*, pages 373–381. Springer, 2006.

[29] Yongkang Wong, Shaokang Chen, Sandra Mau, Conrad Sanderson, and Brian C Lovell. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In *CVPR 2011 WORKSHOPS*, pages 74–81. IEEE, 2011.

[30] Aditya Nigam and Phalguni Gupta. Quality assessment of knuckleprint biometric images. In *2013 IEEE International Conference on Image Processing*, pages 4205–4209. IEEE, 2013.

[31] Unique identification authority of india. `http://www.uidai.gov.in`, 2021.

[32] Malaysia identity card. `https://www.malaysia.gov.my/portal/content/30582`, 2021.

[33] Xuejun Tan, Bir Bhanu, and YingQiang Lin. Fingerprint identification: classification vs. indexing. In *In Proc. of the Conference on Advanced Video and Signal Based Surveillance*, pages 151–156. IEEE, 2003.

[34] Phalguni Gupta, Kamlesh Tiwari, and Geetika Arora. Fingerprint indexing schemes–a survey. *Neurocomputing*, 335:352–365, 2019.

[35] Luc Devroye and Terry J Wagner. 8 nearest neighbor methods in discrimination. *Handbook of Statistics*, 2:193–197, 1982.

[36] Umarani Jayaraman and Phalguni Gupta. Efficient similarity search on multidimensional space of biometric databases. *Neurocomputing*, 452:623–652, 2021.

[37] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[38] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[39] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *International conference on computer vision*, pages 2564–2571. IEEE, 2011.

[40] Zhenhua Guo, Lei Zhang, and David Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE transactions on image processing*, 19(6):1657–1663, 2010.

[41] Wenyi Lin, Kyle Hasenstab, Guilherme Moura Cunha, and Armin Schwartzman. Comparison of handcrafted features and convolutional neural networks for liver mr image adequacy assessment. *Scientific Reports*, 10(1):1–11, 2020.

[42] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[43] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *ICML workshop on unsupervised and transfer learning*, pages 37–49. JMLR Workshop and Conference Proceedings, 2012.

[44] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 539–546. IEEE Computer Society, 2005.

[45] Loris Nanni, Stefano Ghidoni, and Sheryl Brahnam. Handcrafted vs. non-handcrafted features for computer vision classification. *Pattern Recognition*, 71: 158–172, 2017.

[46] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

[47] Casia iris interval image database version 3.0. `http://www.cbsr.ia.ac.cn/english/IrisDatabase.asp`, 2021.

[48] Avantika Singh, Ashish Arora, and Aditya Nigam. Cancelable iris template generation by aggregating patch level ordinal relations with its holistically extended performance and security analysis. *Image and Vision Computing*, 104:104017, 2020.

[49] Z Sun, T Tan, Y Wang, and SZ Li. Ordinal palmprint representation for personal identification. 2005. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

[50] Ajay Kumar. Incorporating cohort information for reliable palmprint authentication. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 583–590. IEEE, 2008.

[51] Lin Zhang, Lida Li, Anqi Yang, Ying Shen, and Meng Yang. Towards contactless palmprint recognition: A novel device, a new benchmark, and a collaborative representation based identification approach. *Pattern Recognition*, 69:199–212, 2017.

[52] Wei Jia, Bin Wang, Yang Zhao, Hai Min, and Hailin Feng. A performance evaluation of hashing techniques for 2d and 3d palmprint retrieval and recognition. *IEEE Sensors Journal*, 20(20):11864–11873, 2020.

[53] Lin Zhang, Lei Zhang, David Zhang, and Hailong Zhu. Ensemble of local and global information for finger–knuckle-print recognition. *Pattern recognition*, 44 (9):1990–1998, 2011.

[54] Ajay Kumar and Yingbo Zhou. Human identification using knucklecodes. In *International Conference on Biometrics: Theory, Applications, and Systems*, pages 1–6. IEEE, 2009.

[55] Pawel Drozdowski, Christian Rathgeb, and Christoph Busch. Computational workload in biometric identification systems: an overview. *IET Biometrics*, 8(6): 351–368, 2019.

[56] Kevin W Bowyer, Karen Hollingsworth, and Patrick J Flynn. Image understanding for iris biometrics: A survey. *Computer vision and image understanding*, 110(2): 281–307, 2008.

[57] Rajiv Mukherjee and Arun Ross. Indexing iris images. In *International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008.

[58] Ravindra B Gadde, Donald Adjeroh, and Arun Ross. Indexing iris images using the burrows-wheeler transform. In *IEEE International Workshop on Information Forensics and Security*, pages 1–6. IEEE, 2010.

[59] Hunny Mehrotra, Banshidhar Majhi, and Phalguni Gupta. Robust iris indexing scheme using geometric hashing of sift keypoints. *Journal of Network and Computer Applications*, 33(3):300–313, 2010.

[60] Yulin Si, Jiangyuan Mei, and Huijun Gao. Novel approaches to improve robustness, accuracy and rapidity of iris recognition systems. *IEEE transactions on industrial informatics*, 8(1):110–117, 2011.

[61] Umarani Jayaraman, Surya Prakash, and Phalguni Gupta. An efficient color and texture based iris image retrieval technique. *Expert Systems with Applications*, 39 (5):4915–4926, 2012.

[62] Pawel Drozdowski, Christian Rathgeb, and Christoph Busch. Bloom filter-based search structures for indexing and retrieving iris-codes. *IET Biometrics*, 7(3):260–268, 2017.

[63] Emad Taha Khalaf, Muamer N Mohammad, and Kohbalan Moorthy. Robust partitioning and indexing for iris biometric database based on local features. *IET Biometrics*, 7(6):589–597, 2018.

[64] Tauheed Ahmed and Monalisa Sarma. Hash-based space partitioning approach to iris biometric data indexing. *Expert Systems with Applications*, 134:1–13, 2019.

[65] Avantika Singh, Pratyush Gaurav, Chirag Vashist, Aditya Nigam, and Rameshwar Pratap Yadav. Ihashnet: Iris hashing network based on efficient multi-index hashing. In *International Joint Conference on Biometrics (IJCB)*, pages 1–9. IEEE, 2020.

[66] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *International Conference on Engineering and Technology (ICET)*, pages 1–6. IEEE, 2017.

[67] Hong-Bo Deng, Lian-Wen Jin, Li-Xin Zhen, Jian-Cheng Huang, et al. A new facial expression recognition method based on local gabor filter bank and pca plus lda. *International Journal of Information Technology*, 11(11):86–96, 2005.

[68] Steve R Gunn. On the discrete representation of the laplacian of gaussian. *Pattern Recognition*, 32(8):1463–1472, 1999.

[69] Muhammad Arsalan, Hyung Gil Hong, Rizwan Ali Naqvi, Min Beom Lee, Min Cheol Kim, Dong Seop Kim, Chan Sik Kim, and Kang Ryoung Park. Deep learning-based iris segmentation for iris recognition in visible light environment. *Symmetry*, 9(11):263, 2017.

[70] Yung-Hui Li, Wenny Ramadha Putri, Muhammad Saqlain Aslam, and Ching-Chun Chang. Robust iris segmentation algorithm in non-cooperative environments using interleaved residual u-net. *Sensors*, 21(4):1434, 2021.

[71] Huimin Huang, Lanfen Lin, Ruofeng Tong, Hongjie Hu, Qiaowei Zhang, Yutaro Iwamoto, Xianhua Han, Yen-Wei Chen, and Jian Wu. Unet 3+: A full-scale connected unet for medical image segmentation. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1055–1059. IEEE, 2020.

[72] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[73] Tania Johar and Pooja Kaushik. Iris segmentation and normalization using daugman's rubber sheet model. *International Journal of Scientific and Technical Advancements*, 1(1):11–14, 2015.

[74] K Chidananda Gowda and G Krishna. Agglomerative clustering using the concept of mutual nearest neighbourhood. *Pattern recognition*, 10(2):105–112, 1978.

[75] Kapoor Akanksha and Singhal Abhishek. A comparative study of k-means, k-means++ and fuzzy c-means clustering algorithms. In *International conference on computational intelligence & communication technology (CICT)*, pages 1–6, 2017.

[76] Shweta Sharma, Neha Batra, et al. Comparative study of single linkage, complete linkage, and ward method of agglomerative clustering. In *International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pages 568–573. IEEE, 2019.

[77] Odilia Yim and Kylee T Ramdeen. Hierarchical cluster analysis: comparison of three linkage measures and application to psychological data. *The quantitative methods for psychology*, 11(1):8–21, 2015.

[78] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(1):86–97, 2012.

[79] Casia iris lamp database. `http://biometrics.idealtest.org/#/datasetDetail/4`, 2021.

[80] Laurens Van Der Maaten. Accelerating t-sne using tree-based algorithms. *The Journal of Machine Learning Research*, 15(1):3221–3245, 2014.

[81] Tee Connie, Andrew Teoh Beng Jin, Michael Goh Kah Ong, and David Ngo Chek Ling. An automated palmprint recognition system. *Image and Vision Computing*, 23(5):501–515, 2005.

[82] Kamlesh Tiwari, Devendra K Arya, GS Badrinath, and Phalguni Gupta. Designing palmprint based recognition system using local structure tensor and force field transformation for human identification. *Neurocomputing*, 116:222–230, 2013.

[83] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[84] Wageeh W Boles and SYT Chu. Personal identification using images of the human palm. In *Conference on Speech and Image Technologies for Computing and Telecommunications*, volume 1, pages 295–298. IEEE, 1997.

[85] Guangming Lu, David Zhang, and Kuanquan Wang. Palmprint recognition using eigenpalms features. *Pattern Recognition Letters*, 24(9-10):1463–1467, 2003.

[86] AW-K Kong and David Zhang. Competitive coding scheme for palmprint verification. In *International Conference on Pattern Recognition*, volume 1, pages 520–523. IEEE, 2004.

[87] Wei Jia, De-Shuang Huang, and David Zhang. Palmprint verification based on robust line orientation code. *Pattern Recognition*, 41(5):1504–1513, 2008.

[88] Wangmeng Zuo, Zhouchen Lin, Zhenhua Guo, and David Zhang. The multiscale competitive code via sparse representation for palmprint verification. In *Conference on Computer Vision and Pattern Recognition*, pages 2265–2272. IEEE, 2010.

[89] Yong Xu, Lunke Fei, Jie Wen, and David Zhang. Discriminative and robust competitive code for palmprint recognition. *IEEE transactions on systems, man, and cybernetics: systems*, 48(2):232–241, 2016.

[90] Yue-Tong Luo, Lan-Ying Zhao, Bob Zhang, Wei Jia, Feng Xue, Jing-Ting Lu, Yi-Hai Zhu, and Bing-Qing Xu. Local line directional pattern for palmprint recognition. *Pattern Recognition*, 50:26–44, 2016.

[91] Gen Li and Jaihie Kim. Palmprint recognition with local micro-structure tetra pattern. *Pattern Recognition*, 61:29–46, 2017.

[92] Dexing Zhong, Yuan Yang, and Xuefeng Du. Palmprint recognition using siamese network. In *Chinese conference on biometric recognition*, pages 48–55. Springer, 2018.

[93] Lunke Fei, Bob Zhang, Wei Zhang, and Shaohua Teng. Local apparent and latent direction extraction for palmprint recognition. *Information Sciences*, 473:59–72, 2019.

[94] Shuping Zhao and Bob Zhang. Joint constrained least-square regression with deep convolutional feature for palmprint recognition. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.

[95] Dexing Zhong and Jinsong Zhu. Centralized large margin cosine loss for open-set deep palmprint recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(6):1559–1568, 2019.

[96] Jane You, Wai-Kin Kong, David Zhang, and King Hong Cheung. On hierarchical palmprint coding with multiple features for personal identification in large databases. *IEEE Transactions on Circuits and Systems for Video Technology*, 14 (2):234–243, 2004.

[97] Wenxin Li, Jane You, and David Zhang. Texture-based palmprint retrieval using a layered search scheme for personal identification. *IEEE Transactions on Multimedia*, 7(5):891–898, 2005.

[98] Ashish Paliwal, Umarani Jayaraman, and Phalguni Gupta. A score based indexing scheme for palmprint databases. In *IEEE International Conference on Image Processing*, pages 2377–2380. IEEE, 2010.

[99] Xiao Yang, Jianjiang Feng, and Jie Zhou. Palmprint indexing based on ridge features. In *2011 International Joint Conference on Biometrics (IJCB)*, pages 1–8. IEEE, 2011.

[100] Ying-Cong Chen, Meng-Hui Lim, Pong-Chi Yuen, and Jian-Huang Lai. Discriminant spectral hashing for compact palmprint representation. In *Chinese Conference on Biometric Recognition*, pages 225–232. Springer, 2013.

[101] Feng Yue, Bin Li, Ming Yu, and Jiaqiang Wang. Hashing based fast palmprint identification for large-scale databases. *IEEE Transactions on Information Forensics and Security*, 8(5):769–778, 2013.

[102] Feng Yue, Haolun Ding, Bin Li, and Xi Chen. Accelerated pop hashing for fast palmprint identification on large-scale databases. In *Chinese Control Conference (CCC)*, pages 11121–11126. IEEE, 2017.

[103] Jingdong Cheng, Qiule Sun, Jianxin Zhang, and Qiang Zhang. Supervised hashing with deep convolutional features for palmprint recognition. In *Chinese Conference on Biometric Recognition*, pages 259–268. Springer, 2017.

[104] Jumma Almaghtuf, Fouad Khelifi, and Ahmed Bouridane. Fast and efficient difference of block means code for palmprint recognition. *Machine Vision and Applications*, 31(6):1–10, 2020.

[105] Xi Chen, Ming Yu, Feng Yue, and Bin Li. Orientation field code hashing: A novel method for fast palmprint identification. *IEEE/CAA Journal of Automatica Sinica*, 8(5):1038–1051, 2020.

[106] Huikai Shao, Dexing Zhong, and Xuefeng Du. Towards efficient unconstrained palmprint recognition via deep distillation hashing. *arXiv preprint arXiv:2004.03303*, 2020.

[107] Jinsong Zhu, Dexing Zhong, and Kai Luo. Boosting unconstrained palmprint recognition with adversarial metric learning. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 2(4):388–398, 2020.

[108] Shuping Zhao and Bob Zhang. Deep discriminative representation for generic palmprint recognition. *Pattern Recognition*, 98:107071, 2020.

[109] Kevin Lin, Huei-Fang Yang, Jen-Hao Hsiao, and Chu-Song Chen. Deep learning of binary hash codes for fast image retrieval. In *IEEE conference on computer vision and pattern recognition workshops*, pages 27–35, 2015.

[110] Chengcheng Liu, Dexing Zhong, and Huikai Shao. Few-shot palmprint recognition based on similarity metric hashing network. *Neurocomputing*, 2021.

[111] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *International Conference on Machine Learning*, volume 2, page 7, 2016.

[112] Ting-Yun Hsiao, Yung-Chang Chang, Hsin-Hung Chou, and Ching-Te Chiu. Filter-based deep-compression with global average pooling for convolutional networks. *Journal of Systems Architecture*, 95:9–18, 2019.

[113] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *International conference on neural information processing systems*, pages 2488–2498, 2018.

[114] Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461, 2003.

[115] Malcolm Slaney and Michael Casey. Locality-sensitive hashing for finding nearest neighbors. *IEEE Signal processing magazine*, 25(2):128–131, 2008.

[116] Marcus D Bloice, Christof Stocker, and Andreas Holzinger. Augmentor: an image augmentation library for machine learning. *arXiv preprint arXiv:1708.04680*, 2017.

[117] S Aranganayagi and K Thangavel. Clustering categorical data using silhouette coefficient as a relocating measure. In *International conference on computational intelligence and multimedia applications (ICCIMA 2007)*, volume 2, pages 13–17. IEEE, 2007.

[118] Xiangyu He, Peisong Wang, and Jian Cheng. K-nearest neighbors hashing. In *Conference on Computer Vision and Pattern Recognition*, pages 2839–2848, 2019.

[119] Haomiao Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Deep supervised hashing for fast image retrieval. In *IEEE conference on computer vision and pattern recognition*, pages 2064–2072, 2016.

[120] Qing-Yuan Jiang and Wu-Jun Li. Asymmetric deep supervised hashing. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[121] Multiple texture information fusion for finger-knuckle-print authentication system. *Neurocomputing*, 188:190 – 205, 2016.

[122] Face recognition using histograms of oriented gradients. *Pattern Recognition Letters*, 32(12):1598 – 1603, 2011.

[123] Damon L Woodard and Patrick J Flynn. Finger surface as a biometric identifier. *Computer vision and image understanding*, 100(3):357–384, 2005.

[124] Ajay Kumar and Ch Ravikanth. Personal authentication using finger knuckle surface. *IEEE Transactions on Information Forensics and Security*, 4(1):98–110, 2009.

[125] Lin Zhang, Lei Zhang, and David Zhang. Finger-knuckle-print verification based on band-limited phase-only correlation. In *International Conference on Computer Analysis of Images and Patterns*, pages 141–148. Springer, 2009.

[126] Zahra S Shariatmadar and Karim Faez. Finger-knuckle-print recognition via encoding local-binary-pattern. *Journal of Circuits, Systems and Computers*, 22(06): 1350050, 2013.

[127] Guangwei Gao, Lei Zhang, Jian Yang, Lin Zhang, and David Zhang. Reconstruction based finger-knuckle-print verification with score level adaptive binary fusion. *IEEE transactions on image processing*, 22(12):5050–5062, 2013.

[128] Peng Fei Yu, Hao Zhou, and Hai Yan Li. Personal identification using finger-knuckle-print based on local binary pattern. *Applied mechanics and materials*, 441:703–706, 2014.

[129] Guangwei Gao, Jian Yang, Jianjun Qian, and Lin Zhang. Integration of multiple orientation and texture information for finger-knuckle-print verification. *Neurocomputing*, 135:180–191, 2014.

[130] K Usha and M Ezhilarasan. Personal recognition using finger knuckle shape oriented features and texture analysis. *Journal of King Saud University-Computer and Information Sciences*, 28(4):416–431, 2016.

[131] Jooyoung Kim, Kangrok Oh, Beom-Seok Oh, Zhiping Lin, and Kar-Ann Toh. A line feature extraction method for finger-knuckle-print verification. *Cognitive Computation*, 11(1):50–70, 2019.

[132] A. Muthukumar and A. Kavipriya. A biometric system based on gabor feature extraction with svm classifier for finger-knuckle-print. *Pattern Recognition Letters*, 125:150 – 156, 2019.

[133] Umarani Jayaraman, Aman Kishore Gupta, and Phalguni Gupta. Boosted geometric hashing based indexing technique for finger-knuckle-print database. *Information sciences*, 275:30–44, 2014.

[134] Weiying Xie, Baozhu Liu, Yunsong Li, Jie Lei, and Qian Du. Autoencoder and adversarial-learning-based semisupervised background estimation for hyperspectral anomaly detection. *IEEE Transactions on Geoscience and Remote Sensing*, 58 (8):5416–5427, 2020.

[135] Xu Yang, Cheng Deng, Feng Zheng, Junchi Yan, and Wei Liu. Deep spectral clustering using dual autoencoder network. In *Conference on Computer Vision and Pattern Recognition*, pages 4066–4075, 2019.

[136] Jacob Goldberger, Shiri Gordon, Hayit Greenspan, et al. An efficient image similarity measure based on approximations of kl-divergence between two gaussian mixtures. In *International Conference on Computer Vision (ICCV)*, volume 3, pages 487–493, 2003.

[137] Yashas Rai and Patrick Le Callet. Visual attention, visual salience, and perceived interest in multimedia applications. In *Academic Press Library in Signal Processing*, volume 6, pages 113–161. Elsevier, 2018.

[138] Fei Wang, Baba C Vemuri, and Anand Rangarajan. Groupwise point pattern registration using a novel cdf-based jensen-shannon divergence. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1283–1288. IEEE, 2006.

[139] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative

neural samplers using variational divergence minimization. In *Advances in neural information processing systems*, pages 271–279, 2016.

[140] Mohammadreza Armandpour, Patrick Ding, Jianhua Huang, and Xia Hu. Robust negative sampling for network embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3191–3198, 2019.

[141] Ayush Jaiswal, Rex Yue Wu, Wael Abd-Almageed, and Prem Natarajan. Unsupervised adversarial invariance. In *Advances in Neural Information Processing Systems*, pages 5092–5102, 2018.

[142] A Muthukumar and S Kannan. K-means based multimodal biometric authentication using fingerprint and finger knuckle print with feature level fusion. *Iranian Journal of Science and Technology. Transactions of Electrical Engineering*, 37 (E2):133, 2013.

[143] Ting Liu, Andrew W Moore, Alexander Gray, and Claire Cardie. New algorithms for efficient high-dimensional nonparametric classification. *Journal of Machine Learning Research*, 7(6), 2006.

[144] Lin Zhang, Lei Zhang, David Zhang, and Zhenhua Guo. Phase congruency induced local features for finger-knuckle-print recognition. *Pattern Recognition*, 45(7): 2522–2531, 2012.

[145] E.R. Henry. *Classification and Uses of Finger Prints*. George Routledge and Sons London, 1900.

[146] Daniel Peralta, Isaac Triguero, Raul Sanchez-Reillo, Francisco Herrera, and José Manuel Benítez. Fast fingerprint identification for large databases. *Pattern Recognition*, 47(2):588–602, 2014.

[147] Chaochao Bai, Weiqiang Wang, Tong Zhao, and Mingqiang Li. Fast exact fingerprint indexing based on compact binary minutia cylinder codes. *Neurocomputing*, 275:1711–1724, 2018.

[148] Kenneth Ko. *User's Guide to NIST Biometric Image Software (NBIS).* NIST Interagency/Internal Report (NISTIR), National Institute of Standards and Technology, Gaithersburg, MD, 2007.

[149] Raffaele Cappelli. Fast and accurate fingerprint indexing based on ridge orientation and frequency. *IEEE Transactions on Systems, Man, and Cybernetics (Cybernetics)*, 41(6):1511–1521, 2011.

[150] Sung-Oh Lee, Yong-Guk Kim, and Gwi-Tae Park. A feature map consisting of orientation and inter-ridge spacing for fingerprint retrieval. In *International Conference on Audio-and Video-Based Biometric Person Authentication*, pages 184–190, 2005.

[151] V. Anand and V. Kanhangad. Pore based indexing for high-resolution fingerprints. In *International Conference on Identity, Security and Behavior Analysis (ISBA)*, pages 1–6, 2017.

[152] Akhil Vij and Anoop Namboodiri. Fingerprint indexing based on local arrangements of minutiae neighborhoods. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 71–76, 2012.

[153] Umarani Jayaraman, Aman Kishore Gupta, and Phalguni Gupta. An efficient minutiae based geometric hashing for fingerprint database. *Neurocomputing*, 137: 115–126, 2014.

[154] Zhe Jin, Andrew Beng Jin Teoh, Thian Song Ong, and Connie Tee. A revocable fingerprint template for security and privacy preserving. *KSII Transactions on Internet & Information Systems*, 4(6), 2010.

[155] Song Wang and Jiankun Hu. Alignment-free cancelable fingerprint template design: A densely infinite-to-one mapping (ditom) approach. *Pattern Recognition*, 45(12):4129–4137, 2012.

[156] Subhas Barman, Samiran Chattopadhyay, Debasis Samanta, Sujoy Bag, and Goutam Show. An efficient fingerprint matching approach based on minutiae to minutiae distance using indexing with effectively lower time complexity. In *International Conference on Information Technology (ICIT)*, pages 179–183, 2014.

[157] Robert S Germain, Andrea Califano, and Scott Colville. Fingerprint matching using transformation parameter clustering. *IEEE Computational Science and Engineering*, 4(4):42–49, 1997.

[158] Bir Bhanu and Xuejun Tan. Fingerprint indexing based on novel features of minutiae triplets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25 (5):616–622.

[159] George Bebis, Taisa Deaconu, and Michael Georgiopoulos. Fingerprint identification using delaunay triangulation. In *International Conference on Information Intelligence and Systems*, pages 452–459, 1999.

[160] Tamer Uz, George Bebis, Ali Erol, and Salil Prabhakar. Minutiae-based template synthesis and matching for fingerprint authentication. *Computer Vision and Image Understanding*, 113(9):979–992, 2009.

[161] M Elmouhtadi, D Aboutajdine, and S El Fkihi. Fingerprint indexing based barycenter triangulation. In *World Conference on Complex Systems (WCCS)*, pages 1–6, 2015.

[162] Xuefeng Liang, Arijit Bishnu, and Tetsuo Asano. A robust fingerprint indexing scheme using minutia neighborhood structure and low-order delaunay triangles. *IEEE Transactions on Information Forensics and Security*, 2(4):721–733, 2007.

[163] Alfredo MuñOz-BriseñO, AndréS Gago-Alonso, and José HernáNdez-Palancar. Fingerprint indexing with bad quality areas. *Expert Systems with Applications*, 40 (5):1839–1846, 2013.

[164] Alfredo Muñoz-Briseño, Andrés Gago-Alonso, and José Hernández-Palancar. Using reference point as feature for fingerprint indexing. In *Iberoamerican Congress on Pattern Recognition*, pages 367–374, 2014.

[165] Ogechukwu Iloanusi, Aglika Gyaourova, and Arun Ross. Indexing fingerprints using minutiae quadruplets. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 127–133, 2011.

[166] Ogechukwu N Iloanusi. Fusion of finger types for fingerprint indexing using minutiae quadruplets. *Pattern Recognition Letters*, 38:8–14, 2014.

[167] Guoqiang Li, Bian Yang, and Christoph Busch. A novel fingerprint indexing approach focusing on minutia location and direction. In *International Conference on Identity, Security and Behavior Analysis (ISBA)*, pages 1–6, 2015.

[168] Raffaele Cappelli, Matteo Ferrara, and Davide Maltoni. Minutia cylinder-code: A new representation and matching technique for fingerprint recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2128–2141, 2010.

[169] Raffaele Cappelli, Matteo Ferrara, and Davide Maltoni. Fingerprint indexing based on minutia cylinder-code. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):1051–1057, 2011.

[170] Yijing Su, Jianjiang Feng, and Jie Zhou. Fingerprint indexing with pose constraint. *Pattern Recognition*, 54:1–13, 2016.

[171] Wei Zhou, Jiankun Hu, Song Wang, Ian Petersen, and Mohammed Bennamoun. Partial fingerprint indexing: a combination of local and reconstructed global features. *Concurrency and Computation: Practice and Experience*, 28(10):2940–2957, 2016.

[172] Pooja A Parmar and Sheshang D Degadwala. A feature level fusion fingerprint indexing approach based on mv and mcc using svm classifier. In *International*

Conference on Communication and Signal Processing (ICCSP), pages 1024–1028, 2016.

[173] Ntethelelo A Mngenge, Linda Mthembu, Fulufhelo V Nelwamondo, and Cynthia H Ngejane. A fingerprint indexing approach using multiple similarity measures and spectral clustering. In *Conference on Computer and Robot Vision (CRV)*, pages 208–213, 2015.

[174] Lihi Zelnik-Manor and Pietro Perona. Self-tuning spectral clustering. In *Advances in neural information processing systems*, pages 1601–1608, 2005.

[175] Guoqiang Li, Bian Yang, and Christoph Busch. A score-level fusion fingerprint indexing approach based on minutiae vicinity and minutia cylinder-code. In *International Workshop on Biometrics and Forensics (IWBF)*, pages 1–6, 2014.

[176] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[177] Kai Cao and Anil K Jain. Fingerprint indexing and matching: An integrated approach. *International Journal for Computational Biology (IJCB)*, 2017.

[178] Craig I Watson and CL Wilson. Nist special database 4. *National Institute of Standards and Technology Fingerprint Database*, 17:77, 1992.

[179] Tong ZHAO Ru-xin WANG Ming-qiang LI Chao-chao BAI, Wei-qiang WANG. Deep learning compact binary codes for fingerprint indexing. *Frontiers of Information Technology and Electronic Engineering*, 1998.

[180] Kameswara Rao et al. Finding the core point in a fingerprint. *IEEE Transactions on computers*, 100(1):77–81, 1978.

[181] Navrit Kaur Johal and Amit Kamra. A novel method for fingerprint core point detection. *International Journal of Scientific & Engineering Research*, 2(4), 2011.

[182] The FVC2002 database:. http://bias.csr.unibo.it/fvc2002/.

[183] Dario Maio, Davide Maltoni, Raffaele Cappelli, Jim L Wayman, and Anil K Jain. Fvc2004: Third fingerprint verification competition. In *International conference on biometric authentication*, pages 1–7. Springer, 2004.

[184] Jie Zhou, Jinwei Gu, and David Zhang. Singular points analysis in fingerprints based on topological structure and orientation field. In *Advances in Biometrics*, pages 261–270. Springer, 2007.

[185] Puneet Gupta and Phalguni Gupta. A robust singular point detection algorithm. *Applied Soft Computing*, 29:411–423, 2015.

[186] Shan Juan Xie, Hyouck Min Yoo, Dong Sun Park, and Sook Yoon. Fingerprint reference point detemination based on a novel ridgeline feature. In *IEEE International Conference on Image Processing (ICIP)*, pages 3073–3076. IEEE, 2010.

[187] Kamlesh Tiwari and Phalguni Gupta. Meandering energy potential to locate singular point of fingerprint. In *International Conference on Biometrics (ICB)*, pages 1–6, 2016.

[188] Yonghong Liu, Baicun Zhou, Congying Han, Tiande Guo, and Jin Qin. A method for singular points detection based on faster-rcnn. *Applied Sciences*, 8(10):1853, 2018.

[189] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[190] Jie Zhou, Fanglin Chen, and Jinwei Gu. A novel algorithm for detecting singular points from fingerprint images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(7):1239–1250, 2009.

[191] Bir Bhanu and Xuejun Tan. Fingerprint indexing based on novel features of minutiae triplets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25 (5):616–622, 2003.

[192] L Liang, H Zhao, P He, and H Tian. Algorithm of extracting core point in fingerprint. *JOURNAL-HEBEI UNIVERSITY OF TECHNOLOGY*, 36(1):46, 2007.

[193] Ilaiah Kavati, Munaga V. N. K. Prasad, and Chakravarthy Bhagvati. *Hierarchical Decomposition of Extended Triangulation for Fingerprint Indexing*, pages 21–40. Springer International Publishing, 2017.

[194] Sanghoon Lee and Ik Rae Jeong. Improved fingerprint indexing based on extended triangulation. *IEEE Access*, 9:8471–8478, 2021.

[195] Naveed Riaz, Ayesha Riaz, and Sajid Ali Khan. Biometric template security: an overview. *Sensor Review*, 2018.

List of research publications

Journal Publications

1. G. Arora, A. Singh, A. Nigam, H.M. Pandey, K. Tiwari, "*FKPIndexNet: An Efficient Learning Framework for Indexing Finger-Knuckle-Print Database for Faster Identification*", Knowledge-Based Systems, Elsevier, vol. 239, pp. 108028, 2022. **[Q1 with Impact Factor 8.039]**

2. G. Arora, S. Kalra, A. Bhatia and K. Tiwari, "*PalmHashNet: Palmprint Hashing Network for Indexing Large Databases to Boost Identification*", IEEE Access, vol. 9, IEEE, pp. 145912-145928, 2021. **[Q1 with Impact factor: 3.367]**

3. U. Jayaraman, P. Gupta, S. Gupta, G. Arora and K. Tiwari, "*Recent development in face recognition*", Neurocomputing, Elsevier, pp.231-245, 2020. **[Q1 with Impact factor: 5.719]**

4. P. Gupta, K. Tiwari, and G. Arora, "*Fingerprint indexing schemes–a survey*", Neurocomputing, Elsevier, pp.352-365, 2019. **[Q1 with Impact factor: 5.719]**

Paper Presentations in Conferences and Workshops:

1. G. Arora, S. Vichare, and K. Tiwari, "*IrisIndexNet: Indexing on Iris Databases for Faster Identification*", **ACM India Joint International Conference on Data Science and Management of Data (CODS-COMAD)**, Banglore, India, pp. 10-18, 2022.

2. G. Arora, T. Aggarwal, and K. Tiwari, "*Softmax coaxial Gaussian track code for fingerprint indexing*", **ACM India Joint International Conference on Data Science and Management of Data (CODS-COMAD)**, Kolkata, India, pp. 326-329, 2019.

3. G. Arora, C.J. Hwang, S. Hira, and K. Tiwari, "*Palmprint Based Human Recognition in Unconstrained Environment*", **International Conference on Computational Science and Computational Intelligence (CSCI)**, Las Vegas, USA, pp. 374-379, 2018.

4. G. Arora, RR Jha, A. Agrawal, K. Tiwari, A. Nigam, "*SP-NET: One shot fingerprint singular-point detector*", **BMVC2019 Workshop on Object Detection and Recognition for Security Screening**, Cardiff, UK, 2019.

Book Chapters:

1. G. Arora, K. Tiwari, and P. Gupta, "*Liveness and Threat Aware Selfie Face Recognition*", **Selfie Biometrics**, Springer, Cham, pp. 197-210, 2019.

2. K. Tiwari, G. Arora, and P. Gupta, "*Indexing for Healthcare Biometric Databases*", **Design and Implementation of Healthcare Biometric Systems**, IGI Global, pp. 55-72, 2019.

3. G. Arora, JC. Joshi, KK. Gupta, and K. Tiwari, "*Indexing on Biometric Databases*", **AI and Deep Learning in Biometric Security**, CRC Press, pp. 257-282, 2021.

Biography of the Candidate

Ms. Geetika Arora received her B.Tech and M.Tech. degree in Computer Science and Engineering from Banasthali Vidyapith, Rajasthan, India. In 2017, she joined as a full-time Ph.D. scholar in the Department of Computer Science and Information Systems at Birla Institute of Technology and Science (BITS) Pilani, India under the supervision of Dr. Kamlesh Tiwari. Her research interest includes Deep Learning, Biometrics and Computer Vision. She has served as a teaching assistant for the courses like Data Mining, Computer Programming, Object Oriented Programming and Data Warehousing at BITS Pilani, Pilani Campus. More about her research contributions can be found out at `shorturl.at/lyFY1`. Contact her at `p2016406@pilani.bits-pilani.ac.in`.

Biography of the Supervisor

Dr. Kamlesh Tiwari is an Assistant Professor in the Department of Computer Science and Information Systems at Birla Institute of Technology and Science Pilani, Pilani campus. He has earned his Ph.D. degree from the Department of Computer Science and Engineering of Indian Institute of Technology Kanpur. His research interests include Machine Learning, Deep Learning, Artificial Intelligence, Computer Vision, Multimodal Biometric (Fingerprint, Face, Palmprint, knuckleprint), and Security. He is an IEEE and Signal Processing Society (SPS) member. He is an APPCAIR affiliated faculty and leads MapMyIndia AI-ML lab. He is a co-incharge of Advanced Data Analytics & Parallel Technologies Lab at BITS Pilani and an active member of Multimedia & HCI Laboratory. More on his research contributions can be found at `www.ktiwari.in`. Contact him at `kamlesh.tiwari@pilani.bits-pilani.ac.in`.