Supervised Deep Learning in Fingerprint Recognition

Chapter · January 2020

DOI: 10.1007/978-981-13-6794-6_7

4 authors, including:

M. Arif Wani

Asif Iqbal Khan
University of Kashmir

Chapter 7

Chapter 7: Application of Supervised Deep Learning in Fingerprint Recognition

7.1. Fingerprint Recognition

Fingerprint recognition refers to the process of identifying or confirming the identity of an individual by comparing two fingerprints. Fingerprint recognition is one of the most researched and reliable biometric technique for identification and authentication. Any system which uses image processing techniques to automatically perform the process of obtaining, storing, analyzing and matching of a fingerprint with another fingerprint and generating the match is called **Automatic Fingerprint Identification System (AFIS)**. It is a system which takes a fingerprint and picks the most likely matches from millions of fingerprint images stored in the database. With the growth in technology, many algorithms and methods have been proposed so far to automatically match the fingerprints without any human interference or assistance.

Figure 7.1: Example of a Fingerprint Image

7.2. Fingerprint Features

A fingerprint consists of a number of ridges and valleys. Ridges are the upper skin layer segments (crest) of the finger and valleys are the lower segments and these ridges run in parallel (Figure 7.1), but there exist one or more regions where they assume distinctive shapes and these regions are called *singularities* or *singular regions/points.* Fingerprint features are divided into three levels level 1 (patterns like singular points), level 2 (minutiae points like bifurcation and ridge endings) and level 3 (pores and contours) features.

Level 1 (pattern) features are the global features which include the general ridge flow and patterns like core and delta location. Singular points are the most important global characteristics of a fingerprint and there are mainly five types of singularities in fingerprints as shown in Figure 7.2.:

- Arch: Ridges form an arc shape which rises at center. (Figure 7.2a)
- Left Loop: Ridges form a curve at center forming a loop towards left. (Figure 7.2b)
- Right Loop: Ridges form a curve at center forming a loop towards the right. (Figure 7.2c)
- Tented Arch: Ridges form an arc, have an angle, an upthrust shape which rises at center. (Figure 7.2d)
- Whorl: Ridges form circularly around a central point. (Figure 7.2e)

Level 2 (minutiae points): In addition to different patterns, fingerprint ridges contain various minute information called minutia points which are very useful in matching fingerprints and are most commonly used in automatic fingerprint matching. Minutia

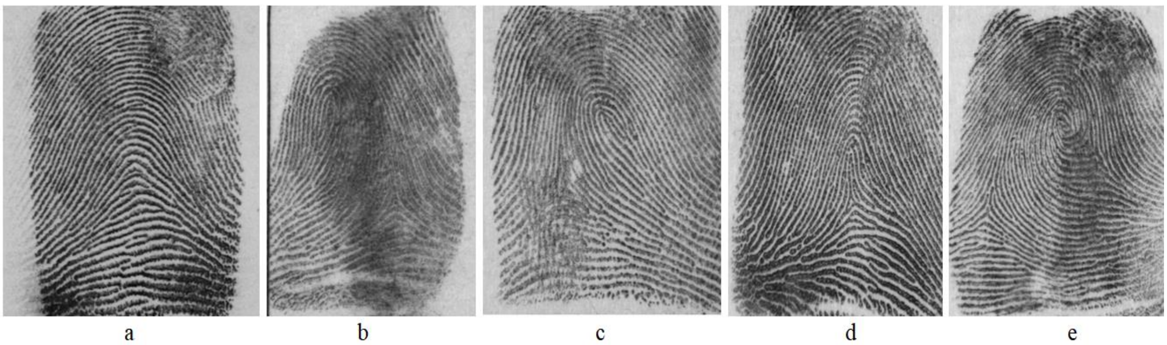


Figure 7.2: Fingerprint classes a) arch b) left loop c) right loop d) tented arch e) whorl

actually means minute details and in context of fingerprints, it means numerous ways a ridge can be disjointed and irregular. There are around 150 types of minutia, out of which ridge ending and ridge bifurcation are most widely used because all other types are the combination of these two.

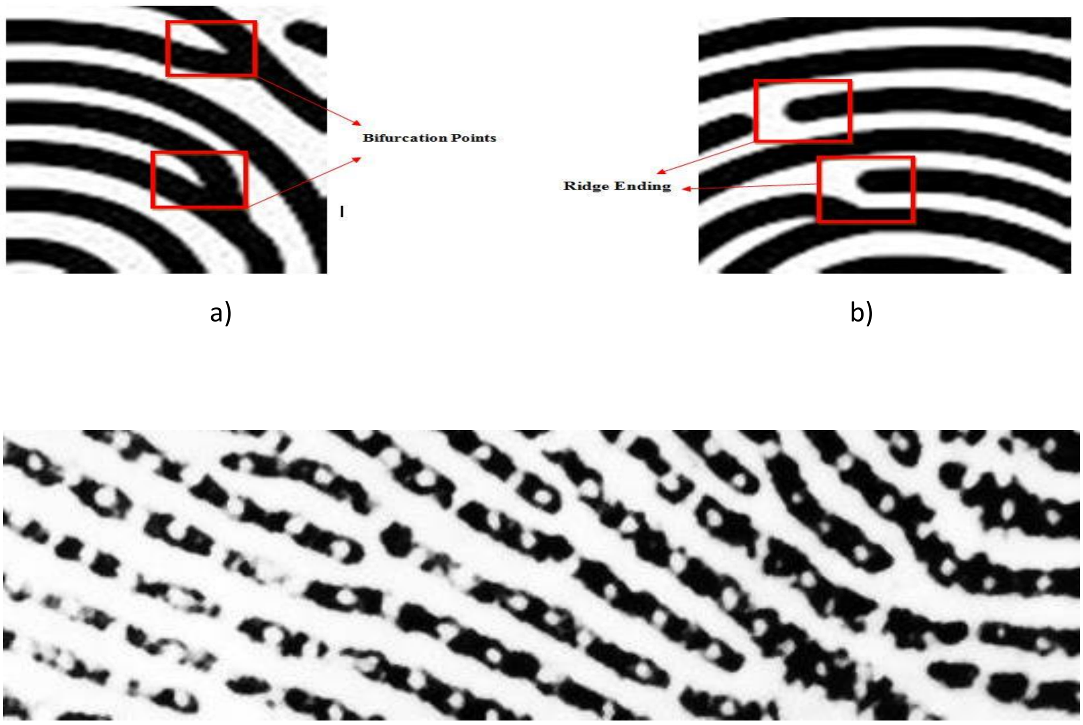
- Ridge Ending: A point where a ridge terminates or suddenly ends. (Figure 7.3a)
- Ridge bifurcation: A point where a ridge divides into two ridges. (Figure 7.3b)

Minutiae points are very important in fingerprints recognition because these points remain unchanged during person's lifetime.

Level 3 (pores and ridge contours) features include all dimensional properties of a ridge, such as sweat pores, ridge edges, width, incipient ridges etc. Extraction of level 3 features from a fingerprint image requires high-resolution images (image resolution of 1000ppi or more) as compared to current standard of 500ppi.

7.3. Automatic Fingerprint Identification System (AFIS)

Minutia based fingerprint matching is the most popular and widely used approach for both human expert and automatic recognition systems. Minutia based approaches have



a) b)

c)

Figure 7.3 a) bifurcation point (level 2 feature), b) ridge ending (level 2 feature) and c) pores, ridge contours (level 3 features) of a fingerprint

many advantages which make them suitable for automatic matching. These approaches require less space because fingerprints are not represented as image but as set of points (minutia points) in two dimensional plane. Minutia points are extracted from two fingerprints and stored as set of points which take very less space as compared to images. Also, matching two sets of points is much faster as compared to matching two images pixel by pixel. A minutia based fingerprint recognition system usually consists of two main stages i) Feature Extraction Stage ii) Fingerprint Matching Stage. Feature extraction stage itself consists of many sub stages which include image segmentation, minutiae and singular point extraction, and classification etc. Figure 7.4 below shows the block map of minutia based fingerprint recognition system.

7.3.1. Feature Extraction Stage

In Feature extraction stage, different useful features such as singular points and minutia points are extracted from the fingerprint image. Ridge bifurcation and ridge ending are the two important minutiae points and almost all the minutiae based fingerprint recognition systems use at least these two minutia types. The accuracy of a fingerprint recognition system depends on the accuracy of feature extraction stage.

Feature extraction stage consists of many steps which include:

- ***Image Enhancement*:**

  The first important thing in fingerprint recognition system is quality of fingerprint image. The performance of an automatic fingerprint recognition system depends a

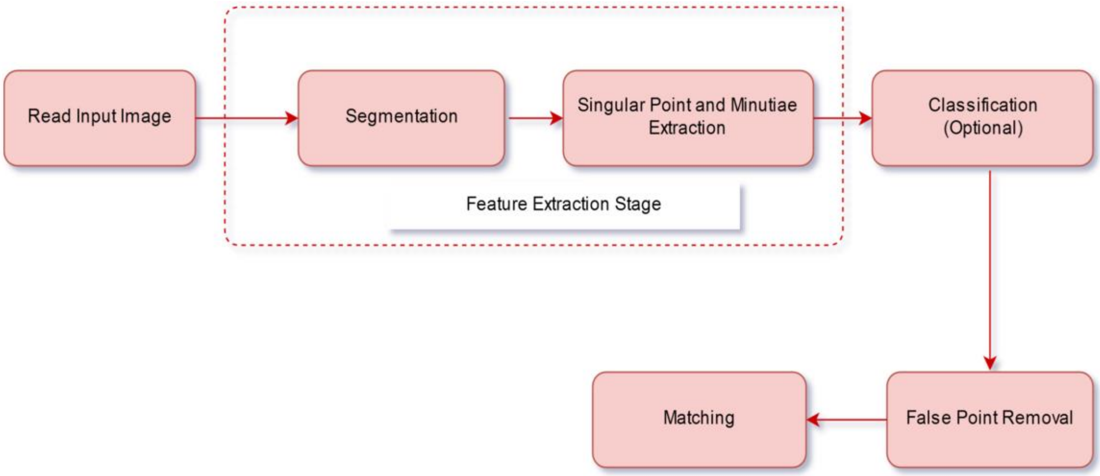


Figure 7.4 Block Map of Automatic Fingerprint Recognition System

lot upon the image quality and the preprocessing steps used. Fingerprint images extracted from various sources usually lack sufficient clarity and contrast. Hence image enhancement is necessary and a major step in AFIS. Image enhancement is used to remove the noise from fingerprint image with the help of various techniques like Histogram, Gaussian smoothing filter etc.

- ***Segmentation:***

  Fingerprint segmentation is the process of extracting Region of Interest (ROI) from a fingerprint image. It is carried out by removing the background from a fingerprint image.

- ***Minutiae Extraction*:**

  After obtaining enhanced segmented image, minutiae extraction is done using the following two main techniques:

  (1) Gray Scale Based Extraction:

  Gray scale based extraction uses ridge tracing in which, starting at a point, ridges are traced by sailing along the ridges in a gray scale image and all the occurrences where a ridge ends or bifurcates are recorded.

  (2) Binarization Based Extraction:

  In binarization based extraction, the fingerprint image is first transformed into 1-bit image with 0-value for ridges and 1-value for furrows or valleys. Binary image consists of only two values 0 for black color and 1 for white color. The grayscale image is transformed to black and white by comparing each pixel value to a threshold value. If the pixel value is lower than the threshold value, the pixel value is assigned black otherwise it is assigned white. The binarized image is then thinned to obtain image with all the ridges just 1 pixel wide followed by minutia points extraction by comparing each pixel along with its neighbouring pixels of the image with predefined specified templates.

- ***False Minutiae Removal:***

Any false minutia points introduced due to inadequate ink, ridge cross-connections, over inking, and due to limitations of minutiae extraction techniques should be deleted for better reliability of the system. Some of these false minutia points are deleted by removing all those points that are too close to each other.

7.3.2. Minutia Matching Stage

After extraction of features from fingerprints, matching is done to obtain the matching score. Fingerprint matching is a challenging task due to variations in the minutia details of different impressions of the same finger. The variations are due to displacement, rotation, non-linear distortion, pressure, skin condition, etc. and as well as due to any errors that may be associated with feature extraction. There are various techniques available for fingerprint matching, some are based on using global features, some use local features and some use both.

7.4. Application of Deep Learning in Fingerprint Recognition

Deep learning, especially convolutional neural network (CNN) has made tremendous success in the field of computer vision and pattern recognition as it does not require handcrafted feature extraction. Deep learning automatically learns features and structures under a sufficient number of training data. These advantages of CNNs makes it suitable for various tasks in automatic fingerprint recognition/identification system: including segmentation, classification, feature extraction (minutiae points and singular points), ridge orientation estimation etc. In next sub sections, application of deep learning in fingerprints recognition will be discussed.

7.4.1. Deep Learning for Fingerprint Segmentation

Fingerprint segmentation is the process of decomposing a fingerprint image into two regions: foreground region also known as Region of Interest (ROI) consisting of fingerprint area and background region, which consists of other irrelevant content like noise etc. Fingerprint segmentation involves demarcation of all the foreground regions accurately in a fingerprint image while discarding all irrelevant contents. Accurate fingerprint segmentation is an important and critical step in automatic fingerprint recognition systems as it affects the reliable feature extraction and eventually the overall performance of the system. Although significant progress has been made on automatic segmentation of plain/rolled fingerprints, latent fingerprint segmentation remains one of the most difficult tasks in automatic latent fingerprint recognition due to poor quality of images and complex background. Latent fingerprints can be present on any surface like glass, cup, newspaper, table etc. and very often these surfaces are not clear or regular, thus making it difficult to extract fingerprint from these surfaces. Their ridge structure is not clean and contains stains, spikes, lines, text etc. thus making the segmenting of foreground regions very difficult. For poor quality images and noisy background, patch based segmentation approach is the preferred over other techniques. Patch based segmentation techniques are computationally expensive and slow but very useful in situations where the input image is distorted and background is noisy.

In patch based segmentation method, segmentation problem is posed as classification problem. Input image is divided into fixed size patches and then these patches are fed to a classifier and only positive patches are assembled to form the segmented image.

Segmentation is done by dividing the fingerprint image into blocks, followed by block classification based on gradient and variance information. The patch based segmentation technique for fingerprint images using Convolutional Neural Networks consists of the following four modules:

- Splitter (S)
- Classifier (C)
- False Patch Normalizer (F)
- Patch Assembler (A)

The Splitter (S) module divides an input image into equal size blocks called patches. The Classifier (C) module uses CNN model which classifies each of these patches into fingerprint and non-fingerprint patches. The False-Patch Normalizer (F) module corrects the misclassified and isolated patches. The Patch Assembler (A) module reassembles these patches and generates a segmented image. The block diagram of the patch based segmentation method is shown in Figure 7.5. The convolution model used for patch classification is trained on both plain and latent fingerprints, thus the technique is suitable for both plain and latent fingerprints.

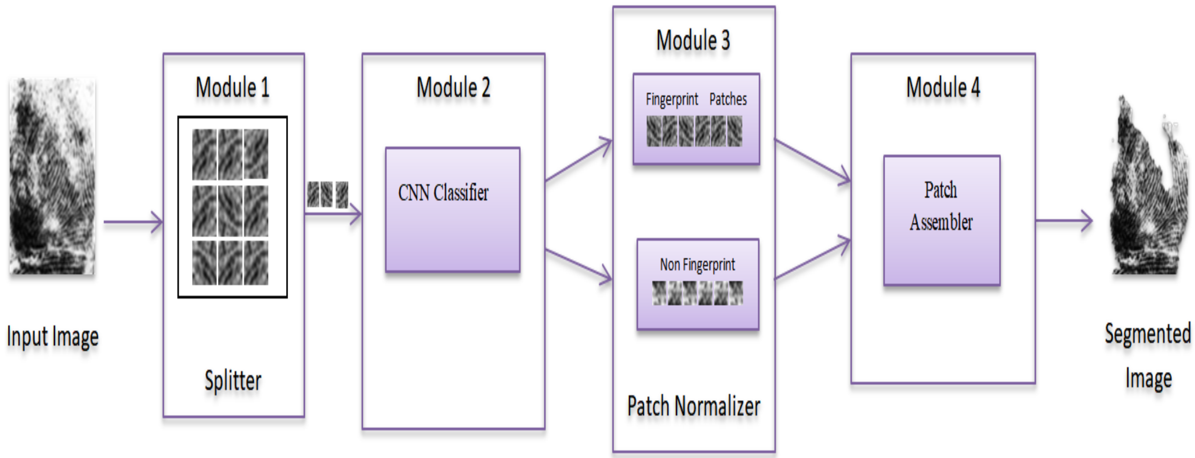


Figure 7.5: Block Diagram of Patch based Fingerprint Segmentation Technique

7.4.1.1. Convolutional Neural Network (CNN) as Patch Classifier

i) CNN Patch Classifier Architecture

Convolutional Neural Network (CNN or ConvNet) based patch classifier is used to classify each patch into fingerprint and non-fingerprint patch. The patch classifier consists of 3 convolutional layers, one subsampling layer, 2 fully connected layers and one dropout layer as shown in Figure 7.6. The output of the last fully-connected layer is fed to a 2-way Softmax classifier which classifies a patch into one of the two types. The first convolutional layer convolves the 16 x16 input image block with 64 filters of size 5 x 5 with a stride of 1 pixel and padding 2 producing 64 feature maps of size 16 x 16. Each convolution layer is followed by a ReLu (Rectified Linear Units) layer. Max pooling is performed after the first convolution operation which produces 16 feature maps of size 8x8. The pooled output of 16 feature maps of size 8x8 is fed to the second

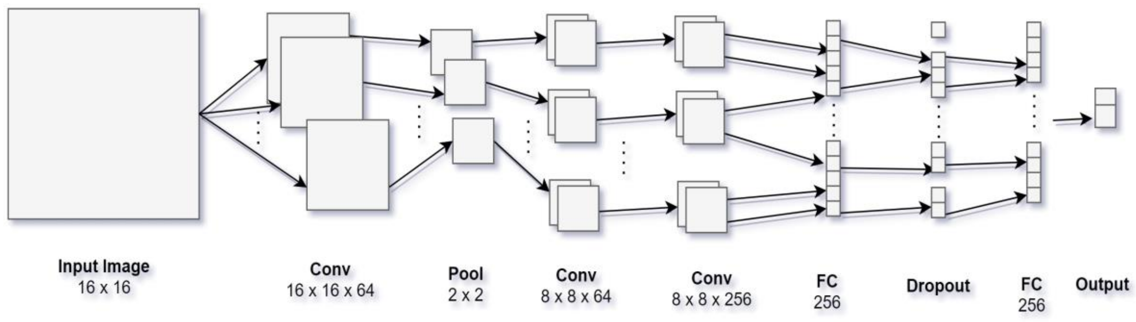


Figure 7.6 Architecture of CNN patch classifier

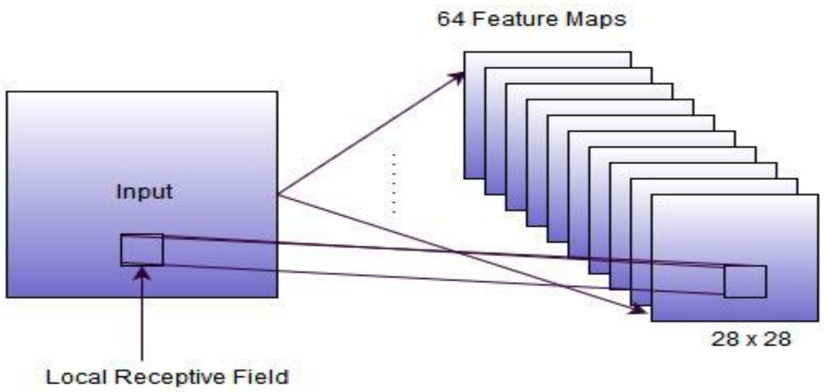


Figure 7.7: First Convolutional Layer

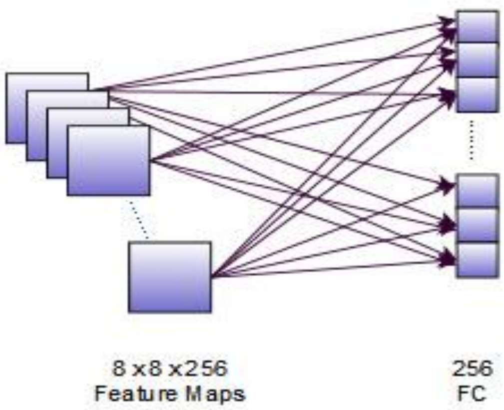


Figure 7.8: Fully connected Layer

convolutional layer which has 64 filters of size 5×5. The second convolution layer produces 16 feature maps of size 8x8. The 16 feature maps of size 8x8 are fed to the third convolutional layer which convolves it with 256 filters of size 5×5. The output is passed to a fully connected layer with 256 neurons followed by dropout layer. Finally, the network has fully connected layer with Softmax classifier producing binary classification. Figures 7.7 and 7.8 shows the operation of first convolution layer and fully-connected layer respectively.

ii) CNN Patch Classifier Training

The CNN patch classifier was trained with around 10000 fingerprint and non-fingerprint patches prepared from IIIT-D latent fingerprint database. Since the training dataset is not sufficient to train CNN model from scratch, the weights were initialized using VGG-16 model. The CNN model was then fine-tuned with 10000 image patches using stochastic gradient descent (SGD) with learning rate of 0.0001, batch size of 50 and weight decay of 1.

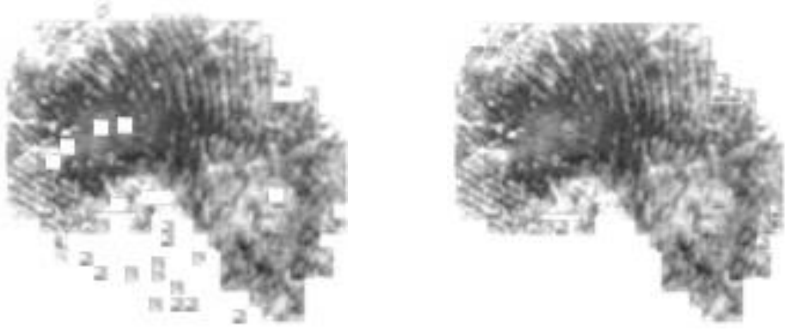
7.4.1.2. False Patch Normalizer

A low score associated with classifying a patch indicates that there is a possibility that the patch may have been misclassified. A misclassified patch is also referred to as a false patch. To reduce the number of false patches, a technique called “majority of neighbors” is used to decide the final label of the patch. Since all the patches of a specific class are likely to be in the same neighborhood, the probability that a patch belongs to the class of majority of its neighbors is high. For each suspicious patch (i.e. whose score is less than a defined threshold), the class label of its four neighboring patches are checked. If at least 3 neighboring patches are of same class as the patch under test then it is accepted as true patch label otherwise it is treated as a false patch and its class label is changed from fingerprint to non-fingerprint or vice-versa depending upon the actual class.

For example, if p_i is i^{th} patch and n_1, n_2, n_3 and n_4 are its four neighbors then

$$label\ (p_i) = majority\ (label(n_1), label(n_2), label(n_3), label(n_4))$$

The method "*Majority of neighbors*" performed well on low quality latent fingerprint images and reduced the false patches by around 20%. The effect of the method is shown in Figure 7.9, which shows segmentation result before and after removing false patches.



a) b)

Figure 7.9: Segmentation result a) before and b) after applying false patch removal technique

7.4.1.3. Patch Assembler

The Patch Assembler takes the classified patches as input, discards negative patches and assembles the positive patches (classified as fingerprint patches) to produce the segmented image. Figure 7.10 shows final result of the segmentation technique on latent fingerprints. Left column contains input images and right column their corresponding segmented image

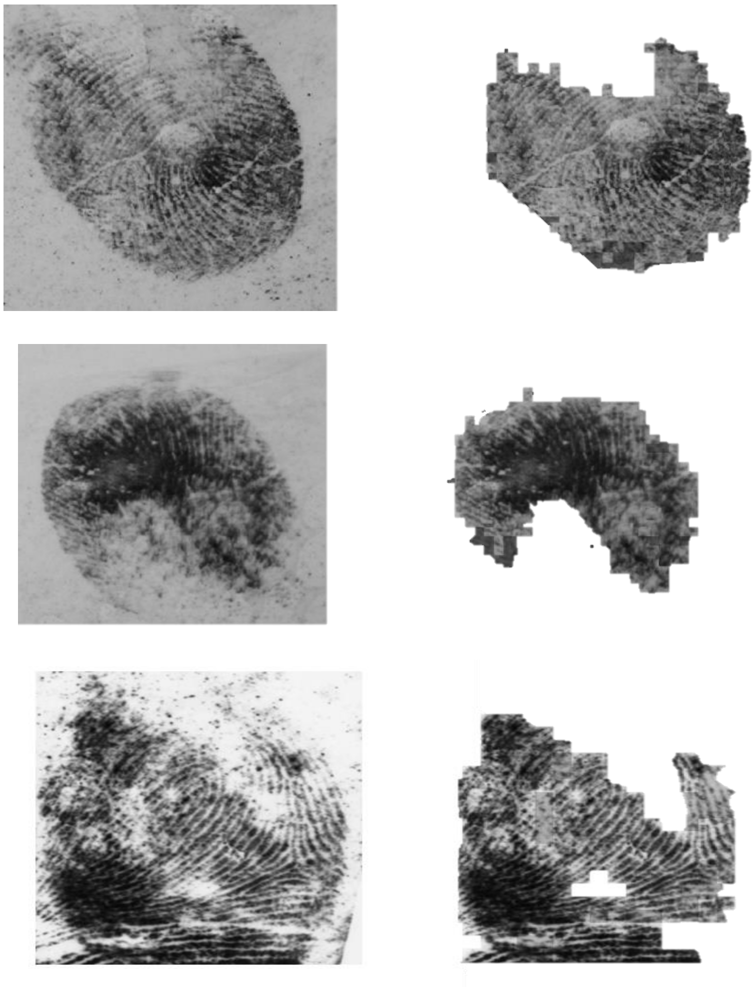


Figure 7.10 *Result of patch based segmentation technique on latent fingerprints from IIIT-D latent database. Left column contains input images and right column their corresponding segmented image*

7.4.1.4. Performance Evaluation

The patch based segmentation technique results are discussed below on IIIT-D latent fingerprint images. To measure the accuracy and precision, following metrics are used:

- **True Positive (TP):** Number of patches correctly classified as belonging to true class.
- **True Negative (TN):** Number of patches correctly classified as not belonging to true class. It is equivalent to correct rejection.
- **False positive (FP):** It is type-1 error and equivalent to false alarm.
- **False Negative (FN):** It is type-1 error and equivalent to miss.

Table I gives the above metrics in the form of confusion matrix.

Table I: Confusion matrix of patch classifier

Total number of images **2346**	True positives (TP) 1022	True negatives(TN) 1065
	False negatives(FN) 196	False positives(FP) 63

The above metrics are used for calculating accuracy, precision and recall as shown below:

Accuracy: It is simply the ratio of correctly predicted observation to the total observations. Mathematical formula is given as:

$$\text{Accuracy} = \frac{TP+TN}{TP+FN+FP+TN} = \frac{2087}{2346} = 88.95$$

Therefore, **Accuracy = 88.95%**

Precision: It is the fraction of relevant instances among the retrieved instances. Formula is given:

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{1022}{1085} = 94.19$$

Precision = 94.19%

Recall: It is the fraction of relevant instances that have been retrieved over total relevant instances in the image. It is based on an understanding and measure of relevance.

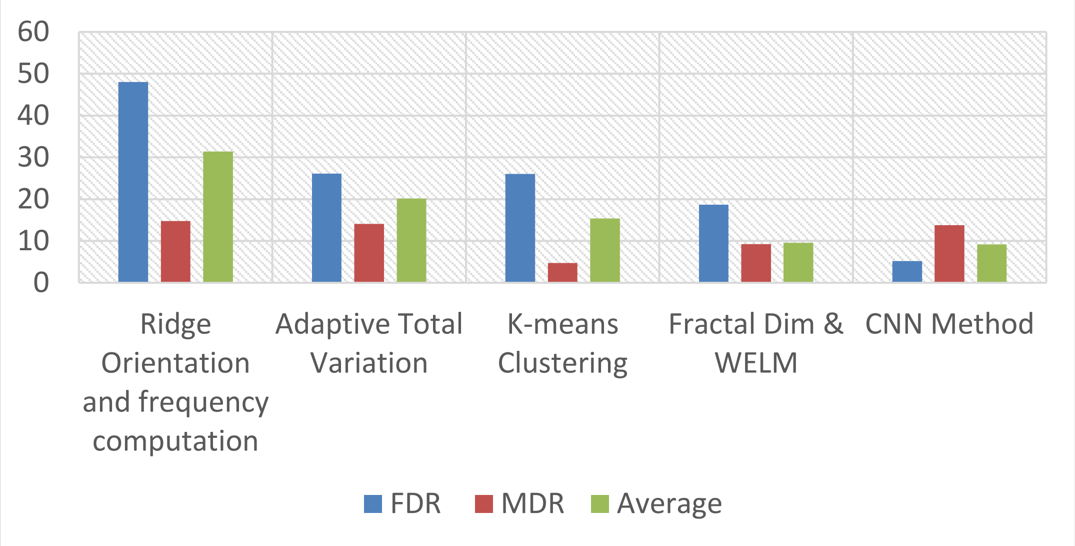
$$\text{Recall} = \frac{TP}{TP+FN} = \frac{1022}{1218} = 83.90$$

Recall = 83.90%

Performance comparison of different segmentation techniques is given in Table II. The table shows false Detection rate (FDR) and Missed Detection Rate (MDR) of different segmentation techniques. The CNN based technique has been tested on IIIT-D latent database only, with FDR of 5.2% and MDR of 13.8%. The second best method tested on the same database has achieved FDR and MDR of 18.7% and 9.22% respectively.

Table II: Performance Comparison of segmentation methods

Approach	Database	FDR %	MDR %	Avg.
Ridge Orientation and frequency computation	NIST SD27	**47.99**	**14.78**	31.38
Adaptive Total Variation	NIST SD27	**26.13**	**14.10**	20.12
K-means Clustering	NIST SD27	**26.06**	**4.77**	15.42
Fractal Dim & WELM	NIST SD27	**18.7**	**9.22**	13.96
	IIIT-D (Good Quality)	**10.07**	**6.38**	8.23
CNN Method	IIIT-D	**5.2**	**13.8**	9.2
	IIIT-D (Good Quality)	**4.7**	**10.5**	7.6



7.4.2. Deep Learning for Fingerprint Classification

Fingerprint classification plays an important role in Automatic Fingerprint Identification System (AFIS) as it effectively reduces the database size as well as matching time. Fingerprints can be broadly divided into five different classes: a) Arch b) Left Loop c) Right Loop d) Whorl and e) Tented Arch. Figure 7.12 shows images of five different fingerprint classes from National Institute of Standards and Technology (NIST) database. Fingerprint classification is one of the important steps in automatic fingerprint recognition systems as it can substantially bring down the number of comparisons at the time of matching.

The CNN based technique, which is called LAFIN classifies a fingerprint into five

a b c d e

Figure 7.12: Fingerprint classes a) arch b) left loop c) right loop d) tented arch e) whorl

different classes and then extracts the singular point from the fingerprint image. It uses a 22-layer CNN model trained on NIST DB4 dataset for classification of fingerprints.

7.4.2.1. CNN Model for Fingerprint Classification

The CNN-model (LAFIN) for classification of fingerprints consists of 22 layers including 5 convolutional layers, 3 pooling layers, 3 fully connected layers and one dropout layer. The output of the last fully-connected layer is fed to a 5-way softmax classifier which classifies the input into one of the 5 labels. The block diagram and architecture of the CNN model are shown in Figure 7.13a and Figure 7.13b respectively.

The first convolutional layer convolves the 224 x 224 x 3 input image block with 64 filters of size 11 x 11 x 3 with a stride of 4 pixels and 0 padding producing 64 feature maps. A ReLu (Rectified Linear Units) layer follows each convolution layer. Max pooling layer follows the first, second and fifth convolutional layers. The second convolutional layer takes as input the normalized and pooled output of the first convolutional layer and convolves it with 256 filters of size 5 × 5. The third convolutional layer takes input from the normalized output of second convolutional layer and convolves it with 256 filters of size 3 × 3. Third and fourth convolutional

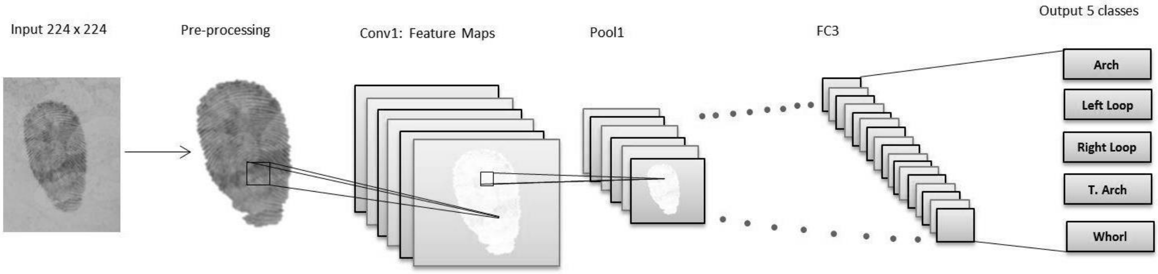


Figure 7.13a: Block diagram of ConvNet model in LAFIN

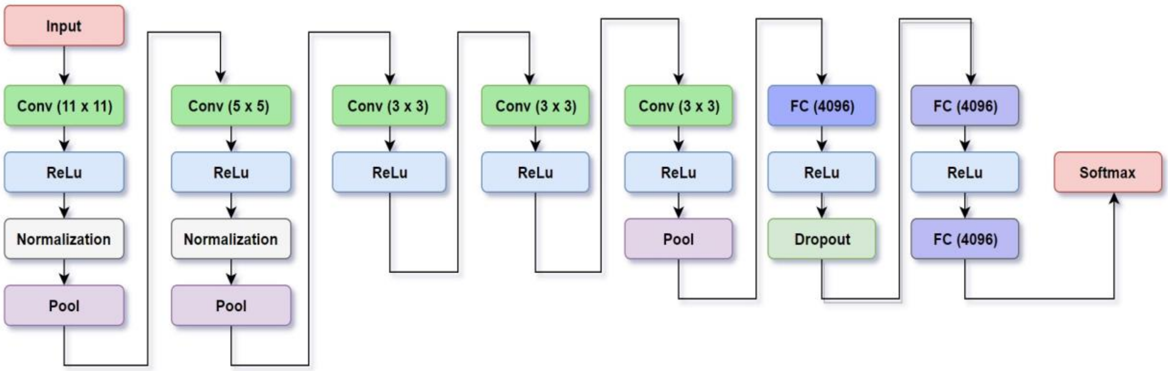


Figure 7.13b: Architecture of ConvNet model in LAFIN

layers are connected without any pooling or normalization layer and both these layers have 256 filters of size 3 x 3. In addition to above mentioned layers, the ConvNet also contains dropout and batch normalization layers to overcome the problem of overfitting. ***Dropout*** refers to dropping out units in a neural network. Dropping a unit means temporarily disconnecting it from the network including all its incoming and outgoing connections. The dropped out units neither contribute to the forward pass nor do they contribute in backpropagation. By using dropout, the network is forced to learn more robust features as network architecture changes with each input.

Batch Normalization

Normalization is the process of scaling down the data to some reasonable limit to standardize the range of features. Batch normalization (BN) is applied after first two convolutional layers to remove covariate shift and reduce training time. Also, it has been observed that BN reduces effects of exploding and vanishing gradients because everything becomes roughly normally distributed.

Batch Normalization is mathematically given as

$$x_i^* = x_i * \frac{M(x)}{\sqrt{V(x)}}$$

where x_i^* is the new normalized value of x_i, $M(x)$ is mean within a batch and $V(x)$ is its variance within a batch. In Convolutional Neural Network, every layer/filter is normalized, i.e. every generated value is treated as a value for normalizing. If the batch size is N and the output (feature map) generated by the convolution has width of W and height of H, then the mean (M) is calculated over N*W*H values (same for the variance). Batch normalization layer follows the first and second convolutional layers. The dropout layer is used after the first fully-connected layer of the CNN model.

7.4.2.2. Performance Evaluation

The CNN model (LAFIN) has been trained on NIST DB4 fingerprint database and IIIT-D latent fingerprint database. NIST DB4 consists of 4000 8-bit grayscale fingerprint image of size 512 x 512 pixels, classified into five classes, Arch (A), Left Loop (L), Right Loop (R), Tented Arch (T) and Whorl (W). IIIT-D Latent Fingerprint database published by Image Analysis and Biometrics lab Indraprastha Institute of Information Technology, Delhi contains 1045 latent fingerprint images from 15 subjects lifted using brush and black powder. The performance of LAFIN CNN model in classifying fingerprint and latent fingerprint images is shown in Table III and Table IV respectively. On IIIT-D latent database, LAFIN achieved classification accuracy of 78.2% while as on NIST-DB4 fingerprint database, the classification accuracy mounted up to 92.2%.

7.4.3. Model Improvement Using Transfer Learning

Deep models like Convolutional neural Network (ConvNet) have large number of parameters that must be learnt before they can be utilized to perform the task of interest. In order to train these models, we require extensively large training dataset with a specific end goal to achieve the desired performance. However, it is relatively subtle to have a dataset of adequate size and due to this reason, it is better not to train the whole

Table III: Classification result of LAFIN Model on IIIT-D Latent fingerprint database

Actual Class	**Predicted Class**					**Accuracy** **78.2%**
	A	L	R	T	W	
A	97	5	2	22	0	76.98%
L	13	107	3	5	9	78.1%
R	11	1	103	5	13	77.4%
T	24	3	5	90	0	73.7%
W	0	11	8	0	109	85.1%

Table IV: Classification result of LAFIN Model on NIST-DB4 fingerprint database

Actual Class	**Predicted Class**					**Accuracy** **92.2%**
	A	L	R	T	W	
A	361	5	2	32	0	90.25%
L	2	376	2	15	5	94.0%
R	8	0	369	18	5	92.25%
T	34	9	5	352	0	88.0%
W	0	8	6	0	386	96.5%

deep network from scratch. Rather, it is normal to pre-train a ConvNet on a vast dataset (e.g. ImageNet, which contains around 1.2 million labelled images with 1000 classifications), and after that use the ConvNet either by fine-tuning the trained model according to the specific task or reuse the learnt weights/parameters in another model for the task of interest. Pre-trained deep models designed for one task can be fine-tuned to achieve better accuracy in other similar tasks. This process of reusing or transferring models learnt for one task into another similar task is called transfer learning. *Transfer learning thus refers to extracting the learnt weights from a trained base network (pre-trained model) and transferring them to another untrained target network instead of training this target network from scratch*. In this way, models learnt in one network are transferred and reused in other network designed to perform similar task. Transfer learning can be used in following ways:

a) ***ConvNet as fixed feature extractor*****:** Here the last fully connected layer (classifier layer) is replaced with a new classifier and this last layer is then trained on new dataset. In this way, the feature extraction layers remain fixed and only the classifier gets fine-tuned. This strategy is best suited when the new dataset is insufficient but similar to the original dataset.

b) ***Fine-tune whole Model*****:** Here a pre-trained model is used with its last fully connected layer (classifier layer) replaced with a new fully connected layer. The whole network is fine-tuned with a new dataset by continuing backpropagation up to the top layers. In this way all the weights are fine-tuned for new task

Results of transfer learning are reported here using three models: two pre-trained models AlexNet and VGG-VD and one model LAFIN trained on two data sets. Alexnet and VGG-VD were fine-tuned for the task of fingerprint classification. Their last 1000-way softmax layer was replaced by 5-way softmax layer (to make the models output five probabilities for five fingerprint classes) and then the fully connected layers were retrained on fingerprint images. LAFIN was first trained on CIFAR-10 dataset and followed by fine-tuning on NIST-DB4. Out of the three models, VGG-VD produced good results. VGG-VD which pre-trained on the large dataset ImageNet (which contains around 1.2 million images with 1000 categories) outperformed the conventional approaches for classification of fingerprints. The classification result of all three models on NIST-DB4 fingerprint database and IIIT-D latent fingerprint database is shown in Table V. Fingerprint classification accuracy of LAFIN improved by around 2% from 92.2% to 94.1% with Transfer Learning. The class wise accuracy of each model is summarized in Table VI, VII and VIII. Out of the three models, VGG-VD produced best results with 95.1% accuracy followed by LAFIN-P with with an accuracy of 94.11%. AlexNet produced results with an accuracy of 93.10%.

The performance improvement after using Transfer Learning are presented in Table IX

Table V: Classification result of the CNN models on NIST DB4 and IIIT-D latent fingerprint databases

Model	Accuracy (NIST-DB4)	Accuracy (IIIT-D)
VGG-VD	95.01%	81.30%
LAFIN	94.11 %	79.75%
AlexNet	93.10%	78.43%

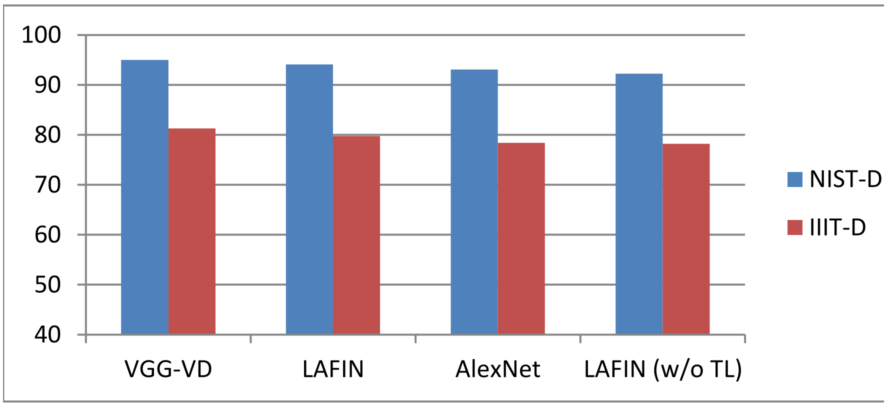


Figure 7.14: Performance of VGG-VD, AlexNet and LAFIN on NIST-DB4 and IIIT-D Latent database

Table VI: Classification result of VGG-VD on NIST-DB4 fingerprint database

Actual Class	Predicted Class A	L	R	T	W	Accuracy 95%
A	**370**	3	2	25	0	92.5%
L	0	**381**	0	13	6	95.3%
R	3	0	**382**	11	4	95.5%
T	19	5	2	**374**	0	93.5%
W	0	5	2	0	**393**	98.25%

Table VII: Classification result of AlexNet on NIST-DB4 fingerprint database

Actual Class	Predicted Class A	L	R	T	W	Accuracy 93.1%
A	**369**	1	0	30	0	92.25%
L	0	**379**	0	17	4	94.75%
R	4	0	**367**	25	4	91.75%
T	24	11	3	**362**	0	90.5%
W	0	7	8	0	**385**	96.25%

Table VIII: Classification result of LAFIN on NIST-DB4 fingerprint database

Actual Class	Predicted Class A	L	R	T	W	Accuracy 94.1%
A	**371**	3	2	24	0	92.75%
L	2	**383**	1	11	3	95.75%
R	7	0	**374**	14	5	93.50%
T	23	8	4	**365**	0	91.25%
W	1	6	4	0	**389**	97.25%

and Table X. Finger print classification accuracy of LAFIN improved by around 2% from 92.2% to 94.1% (see Table X) after Transfer Learning.

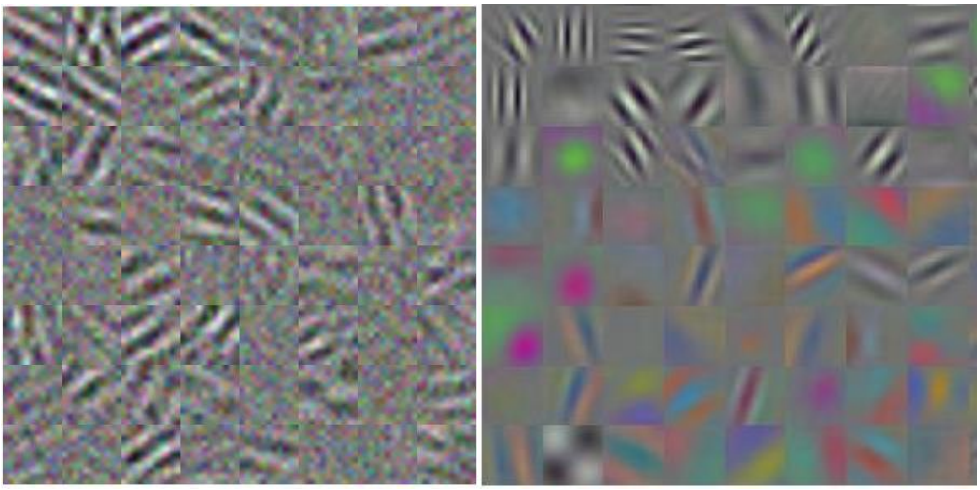
Convolutional neural networks tend to learn first-layer features that either resemble Gabor filters or color blobs. This phenomenon is independent of the network architecture and arises not only for different datasets, but also with very different training objectives. Therefore, these top-level features, which are general for all networks and datasets, are called *general* features. On the other hand, the features learned by the last layer of a trained network must be specific to the chosen dataset and task. These features are specific to a particular task and dataset. Thus, these last layer (bottom level) features are called *specific* features. The features from middle layers (layers between top layer and last layer) show transition from general to specific. For example if a deep network has n layers with 1 being top layer and n being last layer then layer 1 will have maximum generality and minimum or zero specificity and layer n will have minimum or zero generality but maximum specificity. Figure 7.15 below shows first layer features learned by LAFIN before and after Transfer Learning. Filters learned by LAFIN before transfer learning look noisy and incomplete while as features

Table IX: Overall accuracy of LAFIN before and after Transfer Learning

Model	Accuracy (NIST-DB4)	Accuracy (IIIT-D)
LAFIN (Before TL)	92.23%	78.2%
LAFIN-P (After TL)	94.11%	79.75%

Table X: Class wise accuracy of LAFIN before and after Transfer Learning

Actual Class	**Predicted Class**					**Accuracy Before TL**	**Accuracy After TL**
	A	L	R	T	W	**92.2%**	**94.1%**
A	361	5	2	32	0	90.25%	**92.75%**
L	2	376	2	15	5	94.0%	**95.75%**
R	8	0	369	18	5	92.25%	**93.50%**
T	34	9	5	352	0	88.0%	**91.25%**
W	0	8	6	0	386	96.5%	**97.25%**



a) **b)**

Figure 7.15: First layer Filters Learned by LAFIN a) Before and, b) After Transfer Learning

learned after TL look refined.

7.5. Challenges and Future Research Direction

Convolutional Neural Networks can be used at different stages of the fingerprint recognition system. The stages include segmentation, classification and minutiae extraction. Patch based segmentation technique using Convolutional Neural Networks has shown promising results on latent fingerprints. However, the patch-based method is computationally expensive as the input image is divided into a number of small patches and then each patch is fed to a CNN model. If there are 100 patches in an input image, there will be 100 CNN passes for 100 patches thus making the approach both computationally expensive as well as slow. One way to overcome this issue is to use some region based technique to divide the input image into regions (regions likely to contain fingerprint) instead of dividing the entire images into equal sized patches. This will confine fingerprint image processing to those areas that are highly likely to contain useful fingerprints. This can reduce the number of CNN passes to a great extent. Another challenge in automatic fingerprint recognition is extraction of minutiae points from latent fingerprints. A robust CNN based approach to extract minutiae points from a latent fingerprint image can be of great help to forensic experts.

Bibliography

1. Arshad, Irfan, Gulistan Raja, and Ahmad Khan. "Latent Fingerprints Segmentation: Feasibility of Using Clustering-Based Automated Approach." *Arabian Journal for Science & Engineering (Springer Science & Business Media BV)* 39, no. 11 (2014).

2. Cao, Kai, and Anil K. Jain. "Latent orientation field estimation via convolutional neural network." In *Biometrics (ICB), 2015 International Conference on*, pp. 349-356. IEEE, 2015.

3. Cappelli R, Maio D. The state of the art in fingerprint classification. Automatic Fingerprint Recognition Systems. 2004:183-205.

4. Ezeobiejesi, J., & Bhanu, B. (2016). Latent fingerprint image segmentation using fractal dimension features and weighted extreme learning machine ensemble. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 146-154).

5. Guo, W., & Tang, Y. (2013). Latent Fingerprint Recognition: Challenges and Advances. In *Biometric Recognition* (pp. 208-215). Springer, Cham

6. Jain AK, Hong L, Pankanti S, Bolle R. An identity-authentication system using fingerprints. Proceedings of the IEEE. 1997 Sep;85(9):1365-88.

7. Khan AI, Wani MA. Patch-based Segmentation of Latent Fingerprint Images Using Convolutional Neural Network. Applied Artificial Intelligence. 2018 Oct 8:1-5

8. Maltoni, Davide, Dario Maio, Anil K. Jain, and Salil Prabhakar. *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.

9. Mehtre, Babu M., Narayanabhatla Narasimha Murthy, Surendra Kapoor, and B. Chatterjee. "Segmentation of fingerprint images using the directional image." *Pattern recognition* 20, no. 4 (1987): 429-435.