On Latent Fingerprint Minutiae Extraction using Stacked Denoising Sparse AutoEncoders

Anush Sankaran, Prateekshit Pandey, Mayank Vatsa, Richa Singh
IIIT Delhi, India
{anushs,prateekshit12078,mayank,rsingh}@iiitd.ac.in

Abstract

Latent fingerprint identification is of critical importance in criminal investigation. FBI's Next Generation Identification program demands latent fingerprint identification to be performed in lights-out mode, with very little or no human intervention. However, the performance of an automated latent fingerprint identification is limited due to imprecise automated feature (minutiae) extraction, specifically due to noisy ridge pattern and presence of background noise. In this paper, we propose a novel descriptor based minutiae detection algorithm for latent fingerprints. Minutia and non-minutia descriptors are learnt from a large number of tenprint fingerprint patches using stacked denoising sparse autoencoders. Latent fingerprint minutiae extraction is then posed as a binary classification problem to classify patches as minutia or non-minutia patch. Experiments performed on the NIST SD-27 database shows promising results on latent fingerprint matching.

1. Introduction

In law enforcement applications, latent fingerprints are used as a crucial forensic evidence for crime scene analysis. The major steps involved in latent fingerprint matching are: (1) segmenting fingerprint (foreground) from a noisy background, (2) ridge enhancement, (3) extraction of features such as minutiae and singular points, (4) return a list of top-k probable matches (typically $k = 50$) using an Integrated Automated Fingerprint Identification System (IAFIS), and (5) manual verification of the list by forensic experts. Realizing the need for building a "lights-out" IAFIS system, with minimum or no human interference, FBI have launched the Next Generation Identification program [4]. Despite the ongoing research in latent fingerprint matching, feature extraction is still a challenge. As shown in Figure 1, some of the reasons are:

- smudges and strokes introduced by chemical reagents or brush adds to the noise and information loss during latent fingerprint lifting, and
- the surface from which the latent fingerprint is lifted adds to the background noise, thereby making detection of ridge flow challenging.

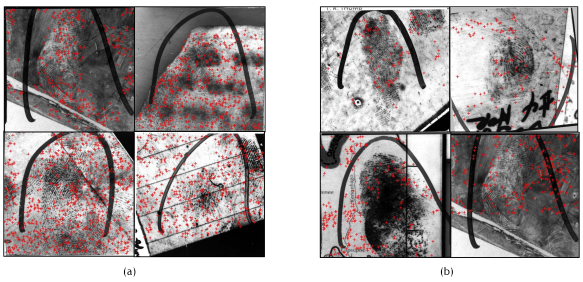

Figure 1. Sample latent fingerprints from NIST SD-27 database [5] showing spurious minutiae extracted by (a) NBIS and (b) VeriFinger 6.0 SDK [3].

Researchers have attempted to address multiple challenges related to latent fingerprint matching [12, 17, 19]. Some of the recent techniques developed to address these challenges are summarized in Table 1. It can be observed that automatic extraction of level-2 features (minutiae) from latent fingerprints is still an open research problem. Specifically, it is challenging to find a robust descriptor to effectively describe or extract minutiae. Some popular minutiae descriptors used in exemplar fingerprints are Minutiae Cylinder Code (MCC) [10], minutiae triplets [21], and Spectral Minutiae Code [23]. In latent fingerprints, Vatsa et al, [20] used Delaunay traingulation to describe the manually annotated level-2 and level-3 features in form of a feature supervector. Paulino et al. [16] used MCC to describe manually annotated minutia neighbourhood. Though MCC is known to provide robust feature description for tenprint fingerprints, it's effectiveness is limited in latent fingerprints due to the following reasons: (i) fewer number of minutiae in latent prints and (ii) it is a completely minutia based descriptor without describing the ridge patterns in the neighbourhood. Recently, Cao et al. [9] proposed to

Paper	Problem addressed	Descriptor used	Remarks
Vatsa et al. [20], 2008	Latent print matching	Delaunay triangulation	Manual annotation of minutiae
Sankaran et al. [18], 2013	Clarity and quality estimation	2-D structure tensor	Manual annotation of minutiae
Paulino et al. [16], 2013	Ridge enhancement	Minutiae Cylinder Code	Manual annotation of minutiae
Cao et al. [9], 2014	Ridge enhancement	Sparse dictionary learning	Automated segmentation of latent fingerprint
Proposed approach	Minutiae detection	Sparse coding using SDSAE	Manual segmentation of latent fingerprint

Table 1. A comparison of different descriptors used in the literature to describe latent fingerprint ridge structure.

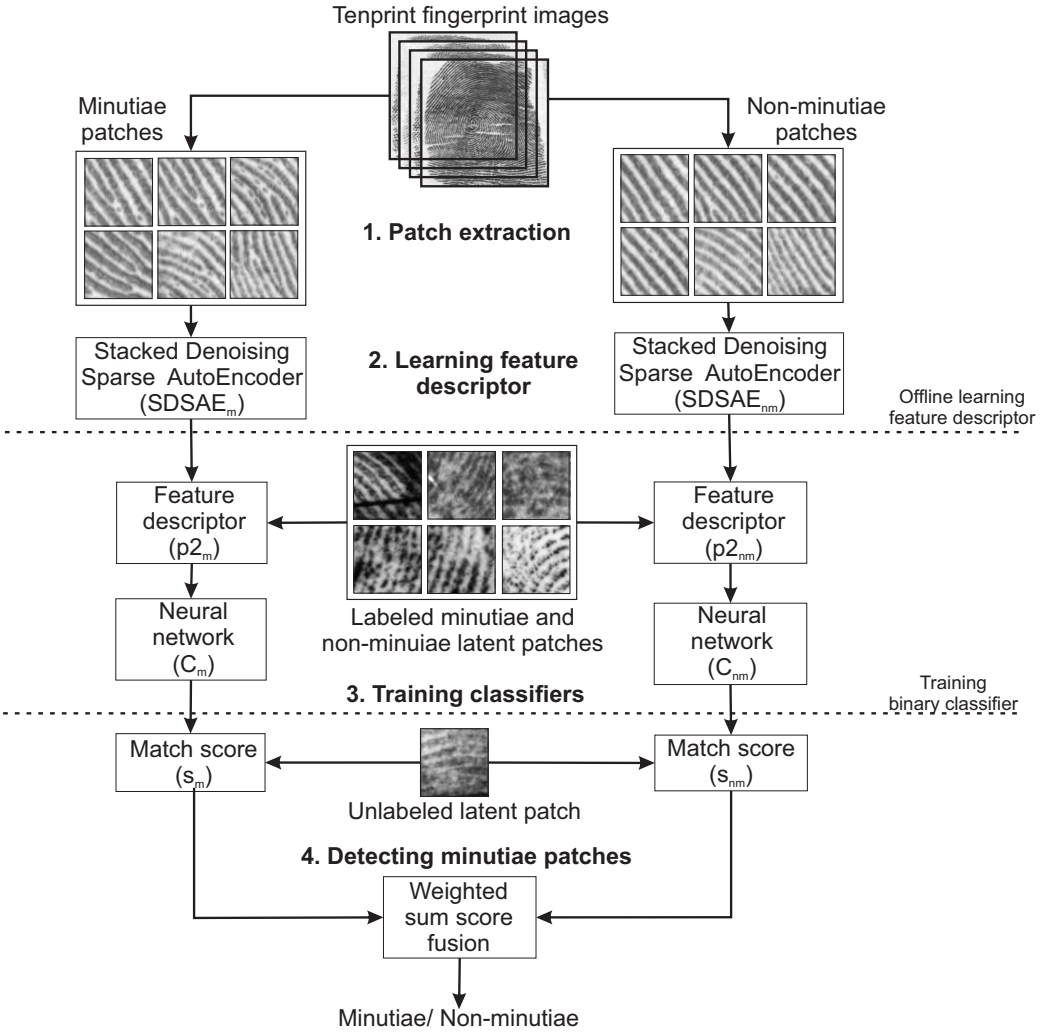

Figure 2. Overall schema of the proposed minutiae detection algorithm using Stacked Denoising Sparse AutoEncoder (SDSAE).

use a sparse dictionary based representation of local ridge structure. Sparse representation encodes both ridge structure and orientation field similarity for a given local fingerprint block. Sparse dictionary learning is combined with adaptive total variation method to perform latent print segmentation and enhancement. However, minutia detection using the learnt representation is still an open research topic.

The aim of this paper is to develop a learning based descriptor for extracting minutiae from latent fingerprints. Adopting from the literature of unsupervised feature learn-

ing, we propose to utilize sparse autoencoders to learn non-linear feature representations of the input data [8]. To learn more complex and robust feature representations, the sparse autoencoders can be stacked in a layer-wise fashion to form a deep network. As latent fingerprints are highly noisy, it is imperative to learn a denoised feature descriptor for minutiae. The key research contributions of this paper are as follows:

1. a novel learning based fingerprint patch descriptor algorithm using stacked denoising sparse autoencoder, and
2. a binary classification model for detecting minutiae in latent fingerprint, using the learned patch descriptor.

2. Proposed Minutiae Detection Algorithm

The key idea in this research is to learn a descriptor for the neighbourhood of a minutiae, i.e. the difference in the structure of patches having and not having a minutiae. The major steps involved in the proposed approach for latent fingerprint minutiae detection are as follows:

1. **Learning feature descriptor:** Minutiae patches and non-minutiae patches are extracted separately from tenprint fingerprint images. The aim is to learn separately a minutiae patch descriptor and a non-minutiae patch descriptor from these local patches using Stacked Denoising Sparse AutoEncoder (SDSAE) [22].
2. **Training binary classifier:** Minutiae extraction in latent prints is presented as a binary classification problem - whether the given latent patch is a minutia patch or a non-minutia patch. Labeled latent print patches (both minutia and non-minutia) are represented using the descriptors learnt in the previous step. Two binary supervised classifiers (one using each descriptor) are then learnt to classify between the minutiae and non-minutiae patches.
3. **Detecting minutiae patch:** Whenever an unseen latent print patch is provided, the minutiae and non-minutiae descriptor of the patch are extracted and classified using both the trained classifiers. The final label is obtained by combining the output of both the classifiers using a weighted sum rule. Minutia extraction in the entire latent print is performed by classifying every local block as a minutiae or non-minutiae patch.

2.1. Learning Feature Descriptor

It can be visually observed from Figure 3, that the neighbourhood of a minutia is different from a normal ridge flow patch. Therefore, a feature descriptor can be learnt that could describe and differentiate a minutia patch from a non-minutia patch. The classical autoencoder is an unsupervised learning algorithm, architecturally similar to feedforward (non-recurrent) neural networks that uses backpropagation algorithm. The learning function produces a non-linear mapping (f_θ) of the input vector $\mathbf{x}$ to a hidden representation $\mathbf{y}$, as follows,

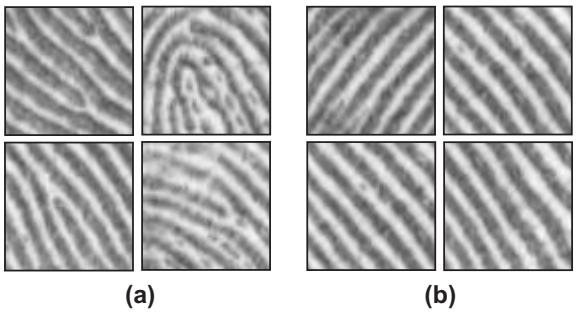

Figure 3. Sample (a) minutiae patches and (b) non-minutiae patches from NIST SD-14 database [7] .

$$\mathbf{y} = f_\theta(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{1}$$

where $s(.)$ is a deterministic sigmoid activation function and $\theta = \{\mathbf{W}, \mathbf{b}\}$ is the set of parameters of the learning function. The hidden representation can be mapped back to the original space using a similar transformation, as follows,

$$\hat{\mathbf{x}} = g_{\theta'}(\mathbf{y}) = s(\mathbf{W}'\mathbf{y} + \mathbf{b}') \tag{2}$$

for the set of parameters $\theta' = \{\mathbf{W}', \mathbf{b}'\}$. Typically, $\hat{\mathbf{x}}$ is an approximate reconstruction of input $\mathbf{x}$ and the autoencoder learns to minimize the reconstruction error between $\mathbf{x}$ and $\hat{\mathbf{x}}$,

$$L_\theta(\mathbf{x}, \hat{\mathbf{x}}) = arg\, min_\theta ||\mathbf{x} - \hat{\mathbf{x}}||_2 \tag{3}$$

A sparse representation of the input data can be obtained by adding a sparsity constraint on the hidden units of the autoencoder, thereby leading to the following optimization function [22]:

$$min \left(\sum_{i=1}^{p} ||\mathbf{x}_i - \hat{\mathbf{x}}_i||_2 + \beta \sum_{j=1}^{m} KL(\rho||y_j) \right) \tag{4}$$

where, p is the size of input data, m is the hidden layer size, β is the weight for the sparsity penalty term, ρ is the constant sparsity level, and $KL(.)$ is the KL-Divergence metric given as,

$$KL(\rho||\hat{\rho}_j) = \rho log \frac{\rho}{\hat{\rho}_j} + (1-\rho) log \left(\frac{1-\rho}{1-\hat{\rho}_j} \right) \tag{5}$$

Denoising sparse autoencoder is an extension of a sparse autoencoder which reconstructs from a partially destroyed or noisy input, $\mathbf{x}'$, as,

$$\mathbf{y} = f_\theta(\mathbf{x}') = s(\mathbf{W}\mathbf{x}' + \mathbf{b}) \quad (6)$$

$$\hat{\mathbf{x}} = g_{\theta'}(\mathbf{y}) = s(\mathbf{W}'\mathbf{y} + \mathbf{b}') \quad (7)$$

However, the reconstruction error is computed as $L_\theta(\mathbf{x}, \hat{\mathbf{x}})$, even though $\hat{\mathbf{x}}$ is reconstructed using $\mathbf{x}'$ and not from $\mathbf{x}$. Thus, the hidden layer is able to learn a robust, denoised representation of the input. A stacked denoising sparse autoencoder is a concatenation of multiple layers of denoising sparse autoencoders, such that the output of the previous layer acts as the input for the next layer [22]. In the greedy learning method, each layer of the stack is learnt independently as a separate denoising autoencoder. In this research, a two layer SDSAE is employed, having a structure as shown in Figure 4.

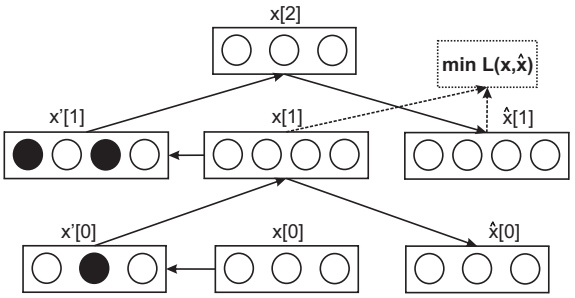

Figure 4. The architecture of SDSAE used in our approach. The input of bottom layers is provided as input to the layer above it.

From a large set of tenprint fingerprint images, n number of minutia patches and equal number of non-minutia patches, each of size $w \times w$, are extracted. Two different autoencoders are utilized ($SDSAE_m$ and $SDSAE_{nm}$), each independently learning the descriptor for a minutia patch and a non-minutia patch, respectively. The stacked autoencoders are of the same architecture and size as $[w^2\ p1\ w^2]$, $[p1\ p2\ p1]$, where $p1$ and $p2$ are the number of hidden nodes in the first and second layers in the stacked autoencoder. The inner-most hidden layer ($p2$), of both the SDSAEs, thus learns a higher level representation of the input patches.

2.2. Training Binary Classifier

As the different layers of the SDSAE are learnt independently, it is essential to stack them together as a single network and fine tune the weights. Fine tuning the weights is also coupled with learning the classification model. It is performed by removing the decoding layers of the autoencoder and adding a classification layer with sigmoid activation function and $L2$ weight decay. The cost function to be minimized is given as

$$L_\theta(x, y) = arg\,min_\theta ||x - y||_2 + \frac{\lambda}{2}\sum_{j=0}^{p2}\theta_j^2 \quad (8)$$

where, λ is the regularization parameter. The additional penalty term makes the above optimization problem strictly convex and guarantees a unique solution. Thus, the binary classifier is a backpropagation neural network of architecture $[w^2\ p1\ p2\ 2]$. Two such classifiers, C_m and C_{nm}, are constructed corresponding to the feature representation extracted from $SDSAE_m$ and $SDSAE_{nm}$. Two independent classifiers are used because each classifier learns to classify minutia and non-minutiae patches using different feature representations. Hence, fusing the two classifiers at decision level or match score level proves to be more useful than concatenating the features. Fine tuning the overall architecture using latent print patches, further assists in adapting the description learnt from tenprint fingerprint patches to the task of latent print patch classification.

2.3. Detecting Minutiae in Latent Fingerprints

A latent print is divided into overlapping patches of size $w \times w$, with a sliding window approach with r pixels overlap. For each patch, $SDSAE_m$ and $SDSAE_{nm}$ are applied to obtain the corresponding descriptors and classification is performed using neural networks, C_m and C_{nm}. The final match score, fs, is obtained by a weighted summation of individual match scores from the two classifiers, and is given as, $fs = \sum_{i=1}^{2} c_i s_i$, where c_i is the weight given to the individual classifiers and s_i is the score of the individual classifiers. The weights, c_i, are calculated as the ratio of the sigmoid activation values of the two nodes in the last classification layer of the neural network, given as $c_i = \frac{a_1^i}{a_2^i}$, such that $a_1^i > a_2^i$, for $i \in 1, 2$. Higher values of c denote a confident classification by the corresponding classifier, and hence, higher weight is provided to the corresponding match score. The final decision label (minutia or non-minutia) is assigned using the score, fs. For all the patches classified as a minutia patch, a minutia point is marked at the center of the patch. It is be noted that the error in minutia point detection ranges between $[0, \frac{w}{2}]$ pixels. It is our assertion that in a bounding-box based minutiae matching algorithm [14], the bounding-box threshold can be accordingly ascertained to mitigate the effect of this error.

3. Experimental Results

3.1. Database

Large number of tenprint fingerprint images are required to learn a robust feature descriptor from SDSAE. Therefore, a heterogenous combined fingerprint database is formed comprising of four publicly available databases: NIST SD-14 v2 [7], CASIA-FingerprintV5 [1], MCYT [15], and FingerPass [13]. Minutiae extraction on all these fingerprint is performed using an open source minutiae extractor MINDTCT of NBIS [2]. The final statistics obtained from

Database	Property	#Images	#Minutiae
NIST SD-14 v2 [7]	Card print database	$54,000$	$8,188,221$
CASIA-FingerprintV5 [1]	Optical sensor	$20,000$	$515,641$
MCYT [15]	Optical, capacitive sensors	$24,000$	$571,713$
FingerPass [13]	Optical, capacitive sensors	$34,560$	$812,643$
Total		**132,560**	**10,088,218**

Table 2. Summary of statistics of the databases in the heterogenous fingerprint database, their properties, number of fingerprint images in each database and the total number of minutiae extracted from the database.

the combined database are shown in Table 2. An image of size 64×64 ($w = 64$) is extracted with minutia as center, thereby creating $10,088,218$ number of minutiae patches. Same number of non-minutiae patches are extracted from every fingerprint to ensure balanced label data while training. This results in $10,088,218$ minutiae and non-minutia patches, independently.

The latent fingerprint database that is used in our experiments is the NIST SD-27 database [5]. The database has 258 latent fingerprints and their corresponding rolled fingerprints. Expert annotated minutiae are provided along with this database. To study the patch descriptor's performance for minutiae extraction, we assume that the latent fingerprints are manually segmented. A random 50% of the data (129 images) are chosen for training. The remaining 129 images are used as testing data, where 64×64 ($w = 64$) patches are extracted with an overlap size of $r = 16$. When a smaller overlap size is used, genuine minutiae is missed because a single patch may have multiple genuine minutiae. When a larger overlap size is used, a single genuine minutiae might be shared over successive patches resulting in spurious minutiae.

3.2. Minutiae Descriptor

The optimal size of the neural network is experimentally found to be [4096 1200 30 2], with the size of hidden layer of stacked autoencoders, $p1 = 1200$ and $p2 = 30$. The other parameter values of SDSAE are chosen to be: the sparsity constant $\rho = 0.05$ and regularization parameter $\lambda = 1$. To visualize the descriptor learnt by the SDSAE, the basis learnt by every single hidden node in the first layer is shown in Figure 5 and Figure 6. It can be seen that each hidden node learns a higher-order sparse representation of the input ridge patterns.

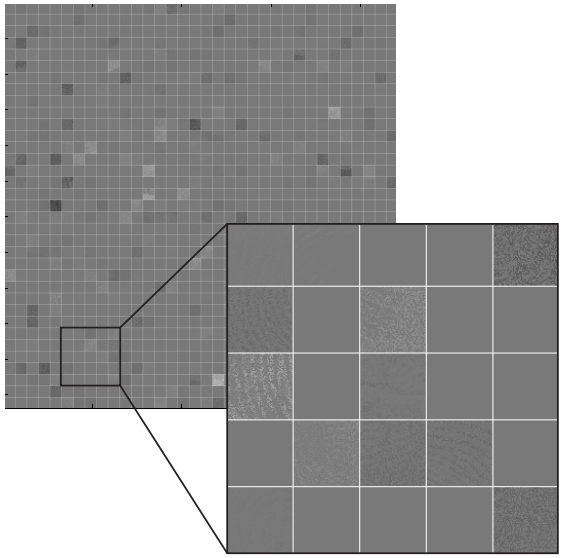

Figure 5. Visualization of all the bases of the $SDSAE_m$. A sparse representation of the minutiae ridge patterns can be observed.

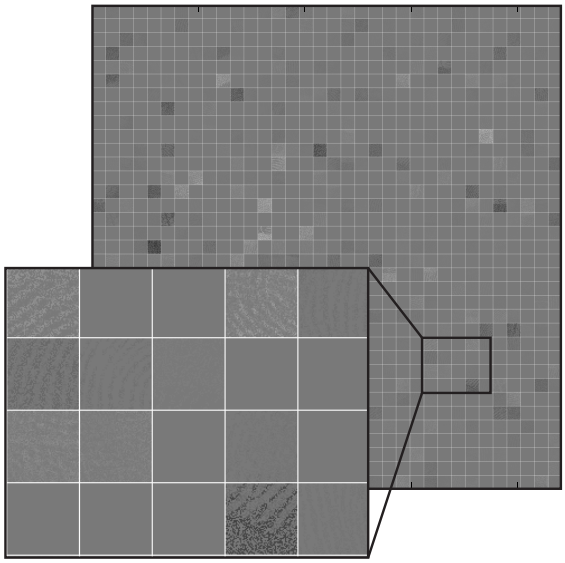

Figure 6. Visualization of all the bases of the $SDSAE_{nm}$. A sparse representation of the non-minutiae ridge patterns can be observed.

3.3. Minutiae Detection Performance

The results predicted by the two classifiers C_m and C_{nm} are combined using weighted sum fusion. The ground truth patch labels are obtained by defining a patch as minutia patch if it contains a manually annotated minutia. The performance of the proposed minutiae extraction algorithm is evaluated using both patch-based metric and minutia-based metric, as follows:

1. **Patch-based metric:** Patch Prediction Accuracy

(PPA) is defined as the ratio of number of patches correctly classified to the total number of patches. This PPA is further subdivided into: Minutia Patch Detection Accuracy (MPDA) and Non-Minutia Patch Detection Accuracy (NMPDA).

- MPDA is defined as the ratio of the number of minutiae patches correctly predicted to the total number of minutiae patches in ground truth.
- NMPDA is defined as the ratio of the number of non-minutiae patches correctly predicted to the total number of non-minutiae patches in ground truth.

An effective minutiae extraction algorithm should have high PPA with high MPDA and NMPDA. Lower values of MPDA will result in loss of actual features, while high values of PPA with low values of NMPDA will result in spurious minutiae.

2. **Minutia-based metric:** The distance between the ground truth minutia set (A) and the predicted minutia set (B) is computed using modified Hausdorff distance [11]. This measure provides an intuition of how much the two minutia sets differ in the 2D-plane, with lesser distance being preferable. The modified Hausdorff distance between two point sets is given as,

$$HD(A,B) = max\left[\frac{1}{|A|}\sum_{a\in A} d(a,B), \frac{1}{|B|}\sum_{b\in B} d(b,A)\right] \quad (9)$$

where $d(a,B) = min_{b\in B}||a-b||$. The average Hausdorff distance is calculated for every predicted minutia set with its corresponding ground truth minutia set.

An average of 8.7 minutiae are extracted per latent fingerprint compared to the average of 12.3 in manually annotated minutiae. Further, on 129 test images of the NIST SD-27 latent fingerprint database, Hausdorff distance of 102.3 is obtained and PPA is observed as 46.8% with MPDA is 65% and NMPDA is 41%. It is observed that C_m classifier has very high threshold in minutia patch detection, while C_{nm} has a lesser threshold producing lots of spurious minutiae. A weighted sum rule of the match scores of both these classifiers produced an optimal MDA of 65%. As the threshold on final score fs increases, MDA increases resulting in a very low NMDA. This changes the PPA from 35% to 68%. The chosen threshold is aimed at maximizing the MDA, thereby reducing the number of missed genuine minutiae. Some false minutiae that are extracted, due to low NMDA, may not affect the matching accuracy because manually marked minutiae are used for the gallery images. Bozorth3 is a robust matcher and hence the false minutiae may not be paired up with any minutiae in gallery image. It is our assertion that due to this reason, higher percent of MDA leads to better matching performance. Also, it is observed that fusion of the two classifiers reduces the Hausdorff distance which suggests that fusion reduces the average distance between ground truth minutiae set and predicted minutiae set. This directly implies the reduced generation of spurious minutiae after classifier fusion, thereby justifying the need for both the feature descriptors.

Algorithm	Rank-10 Accuracy (%)
COTS	16.28
Manual minutiae + bozorth3	26.36
Predicted minutiae + bozorth3	33.61

Table 3. Rank-10 identification accuracy on NIST SD-27 latent fingerprint database using COTS.

3.4. Matching Performance

The aim of a minutia extraction algorithm is to produce good matching performance as in the case of a "lights-out" system. Hence, to evaluate the matching performance of the extracted minutiae, the latent print probe set of 129 minutiae is matched against a gallery set containing 2258 tenprint images, with 258 rolled images from NIST SD-27 and an extended tenprint gallery of 2000 from NIST SD-4 database [6]. The extracted minutiae are matched using the open source matcher bozorth3 from NBIS [2]. The performance of the extracted minutiae are compared with a Commercial Off The Shelf (COTS) matcher[1] and manually annotated minutiae matched with bozorth3 matcher. The rank-10 identification accuracy is shown in Table 3 and the corresponding CMC is plotted in Figure 7. From the results it can be observed that rank 6 onwards the proposed algorithm yields better results than the COTS matcher while rank 8 onwards it yields better performance than the performance of manually annotated minutiae. Rank-1 and rank-2 accuracies are 0% because the average number of minutiae extracted per latent print is lesser than the manually annotated results. Adjusting the classification threshold, fs, will increase the number of minutiae extracted but will also result in an increased number of spurious minutiae. This might lead to false matches. Hence, an optimal threshold, fs, is chosen to trade off between missing genuine minutiae and generated false minutiae.

4. Conclusion and Future Work

The contributions of this paper are two fold: (1) utilizing stacked sparse autoencoders to learn minutiae descriptor and (2) presenting fingerprint minutiae extraction as a

[1]Since there is no publicly available open source matcher for latent fingerprint, we have used a state-of-the-art tenprint matcher

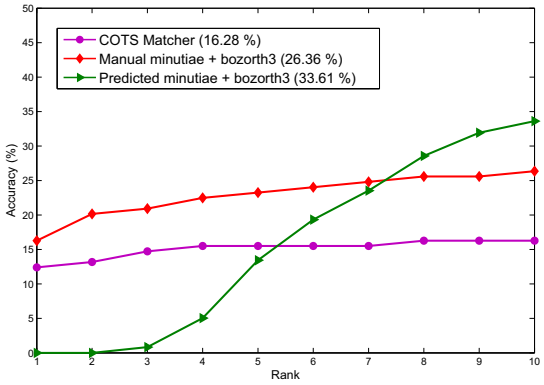

Figure 7. Rank-10 identification accuracy showing comparative results by a COTS matcher, manually annotated minutiae, and predicted minutiae using the proposed algorithm.

binary classification problem and learning a model to classify patches as minutiae or non-minutiae. On NIST SD-27 database, the proposed algorithm improves rank-10 identification accuracy compared to a matching performance using manual feature (minutiae) annotation and commercial system against a gallery of over 2000 identities.

Acknowledgement

Sankaran was partially supported through TCS Research Fellowship.

References

[1] CASIA-Fingerprint V5. *Chinese Academy of Sciences' Institute of Automation (CASIA) Fingerprint Image Database Version 5.0*.

[2] NBIS (NIST Biometric Image Software). *Developed by NIST, http://www.nist.gov/itl/iad/ig/nbis.cfm*.

[3] VeriFinger by NeuroTechnology. *www.neurotechnology.com/verifinger.html*.

[4] FBI, Next Generation Identification. *http://www.fbi.gov/about-us/cjis/fingerprints_biometrics/ngi*, 2008.

[5] Fingerprint minutiae from latent and matching tenprint images. *NIST Special Database 27*, 2010.

[6] Fingerprint minutiae from latent and matching tenprint images. *NIST Special Database 4*, 2010.

[7] NIST 8-bit gray scale images of Fingerprint Image Groups (FIGS). *NIST Special Database 14*, 2010.

[8] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[9] K. Cao, E. Liu, and A. Jain. Segmentation and enhancement of latent fingerprints: A coarse to fine ridge structure dictionary. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2014, doi: 10.1109/TPAMI.2014.2302450.

[10] R. Cappelli, M. Ferrara, and D. 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.

[11] M.-P. Dubuisson and A. K. Jain. A modified hausdorff distance for object matching. In *IAPR International Conference on Pattern Recognition*, volume 1, pages 566–568, 1994.

[12] A. Jain and J. Feng. Latent fingerprint matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):88–100, 2011.

[13] X. Jia, X. Yang, Y. Zang, N. Zhang, and J. Tian. A cross-device matching fingerprint database from multi-type sensors. In *International Conference on Pattern Recognition*, pages 3001–3004, 2012.

[14] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition, 2nd edition*. Springer-Verlag, 2009.

[15] J. Ortega-Garcia, J. Fierrez-Aguilar, D. Simon, J. Gonzalez, M. Faundez-Zanuy, V. Espinosa, A. Satue, I. Hernaez, J.-J. Igarza, C. Vivaracho, D. Escudero, and Q.-I. Moro. Mcyt baseline corpus: a bimodal biometric database. *IEE Proceedings on Vision, Image and Signal Processing*, 150(6):395–401, 2003.

[16] A. A. Paulino, J. Feng, and A. K. Jain. Latent fingerprint matching using descriptor-based hough transform. *IEEE Transactions on Information Forensics and Security*, 8(1):31 –45, 2013.

[17] A. Sankaran, T. I. Dhamecha, M. Vatsa, and R. Singh. On matching latent to latent fingerprints. In *International Joint Conference on Biometrics*, pages 1–6, 2011.

[18] A. Sankaran, M. Vatsa, and R. Singh. Automated clarity and quality assessment for latent fingerprints. In *IEEE International Conference on Biometrics: Theory, Applications and Systems*, pages 1–6, 2013.

[19] M. Vatsa, R. Singh, A. Noore, and K. Morris. Simultaneous latent fingerprint recognition. *Applied Soft Computing*, 11(7):4260–4266, 2011.

[20] M. Vatsa, R. Singh, A. Noore, and S. Singh. Quality induced fingerprint identification using extended feature set. In *IEEE International Conference on Biometrics: Theory, Applications and Systems*, pages 1–6, 2008.

[21] M. Vatsa, R. Singh, A. Noore, and S. K. Singh. Combining pores and ridges with minutiae for improved fingerprint verification. *Signal Processing*, 89(12):2676–2685, 2009.

[22] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2010.

[23] H. Xu, R. Veldhuis, A. M. Bazen, T. A. Kevenaar, T. A. Akkermans, and B. Gokberk. Fingerprint verification using spectral minutiae representations. *Information Forensics and Security, IEEE Transactions on*, 4(3):397–409, 2009.