PATCH BASED LATENT FINGERPRINT MATCHING USING DEEP LEARNING

Jude Ezeobiejesi and Bir Bhanu

Center for Research in Intelligent Systems
University of California at Riverside, Riverside, CA 92521, USA
e-mail: jezeobie@cs.ucr.edu, bhanu@cris.ucr.edu

ABSTRACT

Latent fingerprints are fingerprint impressions unintentionally left on surfaces at a crime scene. Such fingerprints are usually incomplete or partial, making it challenging to match them to full fingerprints registered in fingerprint databases. Latent fingerprints may contain few minutiae and no singular structures. Matching algorithms that entirely rely on minutiae or alignment of singular structures fail when those structures are missing. This paper presents an approach for matching latent to rolled fingerprints using the (a) similarity of learned representations of patches and (b) the minutiae on the correlated patches. A deep learning network is used to learn optimized representations of image patches. Similarity scores between patches from the latent and reference fingerprints are determined using a distance metric learned with a convolutional neural network. The matching score is obtained by fusing the patch and minutiae similarity scores. The proposed system was tested by matching fingerprints segmented from the 258 latent fingerprints in the NIST SD27 database against a database of 2,257 rolled fingerprints from NIST SD27 and SD4 databases. Experimental results show a rank-1 identification rate of 81.35% and highlights the promise of our proposed approach.

Index Terms— Fingerprint, Minutiae, Latent, Matching, Deep Learning, Similarity.

1. INTRODUCTION

Fingerprints have been one of the most reliable methods used in forensics for human recognition. Automated Fingerprint Identification System (AFIS) provides two types of fingerprint searches, namely tenprint search and latent search [9]. Tenprint search involves searching 10 fingers of a person against a fingerprint database of known individuals. Latent search involves searching a fingerprint from a crime scene against a fingerprint database of known individuals [9]. Latent search is used by law enforcement for identifying and apprehending criminals.

Latent fingerprints are usually partial fingerprints and are characterized by few minutiae points, and missing singular points such as core and delta. The dearth of those structures coupled with unspecified orientations, distortions, variations in the illumination of a crime scene, and occlusions, make matching latent fingerprints to full rolled/plain fingerprints very challenging. Many of the existing approaches for matching latent fingerprints to rolled/plain fingerprints rely on fingerprint features mentioned above and become unreliable when the latent fingerprint does not include those structures.

Our approach computes a similarity score by taking into account the overlapped areas on the latent and rolled fingerprints and then matching the minutiae on them, if available. We learn optimal representations of image patches and a similarity function directly from annotated pairs of raw image patches using a deep neural network modeled after a Siamese network (a neural network architecture consisting of two or more identical networks).

2. RELATED WORK AND CONTRIBUTIONS

2.1. Related Work

Many of the existing approaches for matching latent fingerprints rely on features extracted from the latent fingerprints. Jain et al. [9] proposed a latent-to-rolled/plain matching algorithm that relied on manually marked features (minutiae, core, delta) for the latent fingerprints in NIST SD27 database and automatically extracted features for rolled prints in NIST SD4 and NIST SD14 databases. They reported a rank-1 identification rate of 74 percent. Feng et al. [5] used ridge pattern, singular points, and orientation field to match latent fingerprints in NIST SD27 with a database of 10,258 rolled fingerprints. They reported a rank-1 accuracy of 73.3%. Tsai et. al [10] used localized secondary features derived from relative minutiae information and trained a neural network to generate the final similarity score based on minutiae matched in the overlapping areas of a query latent fingerprint and reference fingerprints. They reported 1.21% and 0.68% improvements on minimum total error rates of FVC2002 DB1 and DB2 databases. Deep learning has successfully been applied to latent fingerprint image segmentation [4], and enhancement [11]. Patch-based image matching has been extensively used in computer vision tasks. It has been used for finding accurate correlation between images in domains such as object recognition [12], classification [15], image stitching [1], and image reconstruction [13]. Our approach to patch similarity learning is similar to the techniques used in [6], [10] and [16]. The main difference is that before using a neural network to learn the pairwise similarity between image patches, we first learn optimal representations of the image patches using a pair of deep neural network each consisting of 4 layers of RBMs. To the best of our knowledge, this work represents the first attempt at performing latent fingerprint matching based on learned similarity of image patches.

2.2. Contributions

Our contributions include a new system for patch-based latent fingerprint matching using deep neural networks with an improvement on the previous latent fingerprint matching results. The proposed system learns optimal patch representation and patch similarity without relying on hand crafted features.

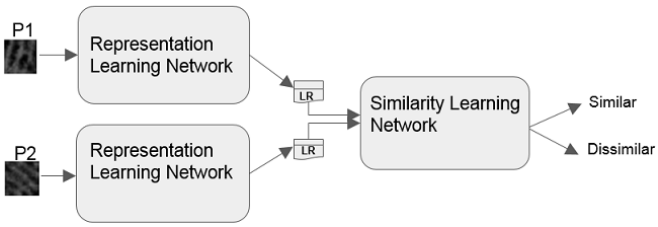

Fig. 1: Proposed architecture for learning fingerprint patch representation and similarity between fingerprint patches. Each Representation Learning Network (RLN) consists of 4 layers of Restricted Boltzmann Machine (RBM). The Similarity Learning Network (SLN) consists a 7-layer deep convolutional network whose components are shown in 2.

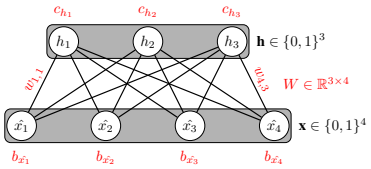

Fig. 2: Graphical depiction of RBM with binary visible and hidden units. $\hat{x_i}, i = 1, \ldots, 4$, are the visible units while $h_k, k = 1, \ldots, 3$, are the hidden units. $b_{\hat{x_i}}, i = 1, \ldots, 4$, are the biases for the visible units and $c_{h_k}, k = 1, \ldots, 3$, are the biases for the hidden units.

3. TECHNICAL APPROACH

The block diagram of our proposed approach is shown in Figure 1. It consists of two neural networks, one for learning patch representations and the other for learning the similarity between the learned representations. During training, a pair of the representation learning network (RLN) similar to Siamese network, but without shared parameters, is applied to pairs of patches (P1, P2). The learned representations (LR) from the RLNs are concatenated to form the input to the similarity learning network (SLN). Similarity learning is supervised while representation learning is unsupervised to facilitate discovery of hidden and interesting patterns in the patches. The RLN and SLN are jointly trained on patch-pairs generated from segmented Latent fingerprint images and rolled images from the NIST SD27 database. Table 1 shows the hyper-parameters and values used to train the RLN. The structure of the SLN is depicted in Table 2. The parameters for RLN were selected based on the performance of the network on the validation set. The choice of 32x32 patch size is based on its empirically determined optimality.

3.1. Architecture and Hyper-Parameters

A Restricted Boltzmann Machine is a stochastic neural network that consists of visible layer, hidden layer and a bias unit [7]. A sample RBM with binary visible and hidden units is shown in Figure 2. RBM has no intra-layer connections and given the visible unit activations, the hidden unit activations are mutually independent. Also the visible unit activations are mutually independent given the hidden unit activations [2].

We selected the value of the hyper-parameters used in the proposed network based on the performance of the network on the validation set. The parameters and values are shown in Table 1

We tried five different networks (with different number of layers and neurons in each layer) in the RLN and chose the architecture that gave the best performance in terms input reconstruction errors. The architectures and their performance on the training and validation datasets are shown in Table 3. The chosen RLN architecture is highlighted in bold.

Table 1: Parameters and values for the representation learning network (RLN). L_i refers to layer i. L_1 is the input layer. Layers 1, 2, and 3 are RBM layers.

Parameter	L_0	L_1	L_2	L_3	L_3
Number of Neurons	1024	1800	1000	1200	1024
Batch Size	-	32	32	32	32
Epochs	-	50	50	50	50
Learning Rate	-	1e-3	5e-4	5e-4	5e-4
Momentum	-	0.70	0.70	0.70	0.70

Table 2: Parameters for the similarity learning network (SLN). Epochs : 100, Batch size : 64, Learning rate : 0.0100, Momentum : 0.9000.

Name	**Type**	**Dim.**	**Filter**	**Stride**
Input	input	32x32x1	-	
Conv	Convolution		30 5x5	
ReLU	ReLU			
MaxPool	Max Pooling	1x1		[1 1]
Softmax	Softmax			
Output	Similarity output			

Table 3: Five candidate architectures and model performance. The difference between the architectures is the number of layers for patch representation learning. The architecture with 5 layers gave the best performance in terms of mean square reconstruction error (MSRE), and was used in this paper. As can be seen from the table, the reconstruction error improved as we added more pre-training layers. The gains started to fade after 5 layers. This result is consistent with the observation in [3] that for certain tasks, going deep improves network performance but after a certain depth, the benefit starts to disappear.

Architecture	No. of layers	Minimum MSRE	Maximum MSRE
Arch1	1	0.0350	0.2180
Arch2	3	0.0476	0.0952
Arch3	**5**	**0.0045**	**0.0562**
Arch4	6	0.0138	0.1085
Arch5	7	0.0248	0.1105

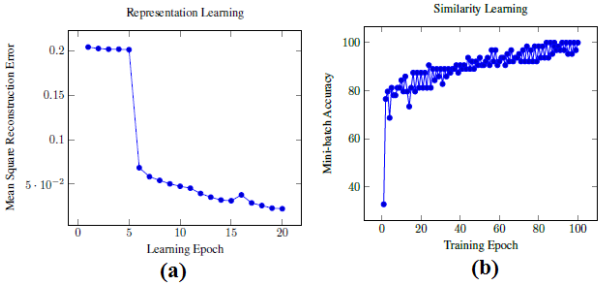

Fig. 3: (a) Plot showing mean square reconstruction error (MSRE) during the training of the RLNs. (b) Plot showing mini-batch accuracy during the training of the SLN.

3.2. Image patch similarity

In this work, we consider two patches to be similar if there exist an in-plane rotation by d degrees that makes them identical. When a given patch p is rotated by $d \in \{0^o, 45^o, 90^o, 135^o, 180^o, 225^o, 270^o, 315^o\}$, we obtain a set of patches $p_r = \{p_1, p_2, \cdots, p_8\}$. Each pair of patches $(p_i, p_k) \in p_r$ are considered similar. This definition of similarity allows us to learn patch representations

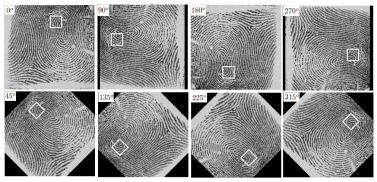

Fig. 4: Various rotations ($0^o, 45^o, 90^o, 135^o, 180^o, 225^o, 270^o, 315^o$) of a sample fingerprint with a 32x32 patch highlighted with a bounding box. By our patch similarity definition, all the patches are similar.

that are invariant to rotations. With this strategy, given an image patch from a reference fingerprint image, the chances of finding a matching patch from the latent fingerprint being matched against it is enhanced. Figure 4 shows in-plane rotations of a sample fingerprint with a 32x32 patch highlighted with a bounding box. The 32x32 patches in all the rotations are similar.

4. MATCHING

4.1. Patch Similarity

Let *L* and *R* be latent and reference fingerprints, respectively. Let $P_L = \{l_1, \cdots, l_k\}$ and $P_R = \{r_1, \cdots, r_n\}$ denote the 32x32 patches from *L* and *R*. For each $l_i \in P_L, i = 1, \ldots, k$, we create tuples $(l_i, r_j), r_j \in P_R, j = 1, \ldots, n$, feed each tuple to our trained model and record the similarity score. To show that using a brute-force matching strategy is feasible in this scenario, we note that a reference fingerprint in the NIST SD27 database is 800x768 while one of the largest segmented fingerprint from a latent fingerprint in the same database is 380x448. The number of 32x32 non-overlapping patches from the reference and latent fingerprints are $\frac{614,400}{1,024} = 600$ and $\frac{170,240}{1,024} = 166$, respectively. Evaluating a match between them requires $166 \times 600 = 99,600$ comparisons in the worst case. To minimize the computation time, a pipeline with 166 parallel computations can be used. We define the patch similarity score for each $l \in P_L$ as:

$$S_l = max(s(l, r_1), \ldots, s(l, r_n)) \tag{1}$$

where $s(l, r_k)$ is the similarity score of tuple $(l, r_k), l \in P_L, r_k \in P_R, k = 1, \ldots, n$. A patch $l \in P_L$ is said to have a match if $S_l \geqq \rho$, where ρ is an empirically determined threshold value (0.75). The patch similarity score between *L* and *R* is defined as:

$$S^p_{LR} = \frac{L_m}{|P_L|} \tag{2}$$

where L_m is the number of patches in *L* with matching patches in *R*, and $|P_L|$ is the total number of patches in *L*.

4.2. Minutiae Matching

Minutiae extraction from patches in *L* with matching patches in *R* was done using minutiae extraction algorithm based on crossing number [14]. The minutiae similarity score is calculated as:

$$S^m_{LR} = \frac{\sigma}{\tau} \tag{3}$$

where σ is the number of patches in *L* with matching minutiae in *R*, and τ is the total number of minutiae in *R*.

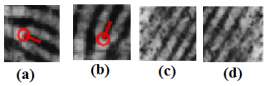

Fig. 5: Examples of matching query and reference fingerprint patches. (a) and (b) are matching patches with minutiae, while (c) and (d) are matching patches without minutiae.

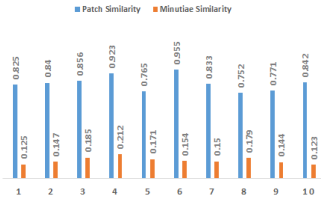

Fig. 6: Sample patch and minutiae similarity scores for 10 segmented latent fingerprints from NIST SD27.

The fused score is given by:

$$F_{mp} = \frac{1}{2}(S^m_{LR} + S^p_{LR}) \tag{4}$$

A threshold of 0.45 was used for F_{mp}. Figure 5 shows examples of matching patches with minutiae and matching patches without minutiae, while Figure 6 shows patch and minutiae similarity scores computed for 10 segmented latent fingerprints.

5. EXPERIMENTS AND RESULTS

We implemented the algorithms in Matlab R2017a running on Intel Core i7 CPU with 8GB RAM and 750GB hard drive. Our implementation relied on NNBox, a Matlab toolbox for neural networks. The implementation uses backpropagation, contrastive divergence, Gibbs sampling, and hidden units sparsity based optimization techniques.

5.1. Training Dataset

The training dataset was created from fingerprint impressions segmented from the latent fingerprint images in the NIST SD27 database, as well as the matching rolled fingerprint images. The segmentation was done using a different deep learning model details of which are omitted for brevity. We split the fingerprint images into 32x32 non-overlapping patches and created 7 similar patches for each patch by rotating the patch by $45^0, 90^0, 135^0, 180^0, 225^0, 270^0$, and 315^0. The training dataset was then partitioned into groups of (p_i, p_j, z) where $z = 1$ if $p_i = p_j$ and 0 otherwise. The operator $=$ reflects patch similarity as defined in section 3.2. A total of 100,000 patches ($\frac{602 \times 380 \times 448}{32 \times 32}$) from 602 fingerprints (150 segmented latents (NIST SD27), 150 rolled fingerprints (NIST SD27), 302 fingerprints (NIST SD4 database) patches were augmented with the rotated patches to obtain a training set containing $100,000 \times 8 = 800,000$ patches.

5.2. Evaluation Datasets

We created two evaluation datasets: query dataset and reference dataset. Regions-of-interest were segmented from the 258 latent fingerprints in NIST SD27 database and saved in a query dataset. The reference dataset containing 2,257 rolled prints was created

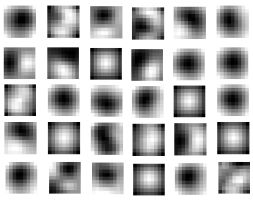

Fig. 7: The 30 5 x 5 filters learned in the convolutional layer of SLN on the 32 x 32 patches from segmented fingerprint impressions in NIST SD27 database. The dark colors represent intensity.

from 2,000 fingerprint images in NIST SD4 database, and the 257 rolled images in NIST SD27 database. We matched each fingerprint in the query dataset against the images in the reference dataset to evaluate the proposed approach.

5.3. Training, Validation and Testing

One set of fingerprint patches at a time from the training dataset was fed to the RLN after being preprocessed to have 0 mean and unit standard deviation. The training of the RLN was done using standard stochastic gradient descent and back-propagation with a batch size of 64 and different learning rates for each layer. In our implementation, we randomly generated the connection map between the visible layer and the first hidden layer. The input weight of all other hidden layers are the output weights of the respective preceding layers. We kept adjusting the learning rate until the mean square reconstruction error stabilized. For the SLN, the objective was to minimize the cross-entropy error defined as:

$$C = -\frac{1}{m}\sum_{k=1}^{m} y_k \log \hat{y}_k + (1 - y_k)\log(1 - \hat{y}_k) \qquad (5)$$

over a training set of m patch pairs using stochastic gradient descent with a batch size of 64. In the above equation, y_k, is the similar(1) or dissimilar(0) label for input pair x_k, while $\hat{y}_k$ and $1 - \hat{y}_k$ are the Softmax activations computed on the values at the two output nodes. We sampled equal number of positive and negative pairs for training to prevent over-fitting. A positive entry in the training dataset is of the form $(p_i, p_k, 1)$, with $p_i = p_k$, while a negative sample is of the form $(p_i, p_j, 0)$, with $p_i \neq p_j$. We used 100,000 positive pairs and 100,000 negative pairs to train the entire network. 40,000 positive and negative pairs were used to validate the network, while 20,000 positive and negative pairs were used for testing. There was no overlap between the training, validation and test datasets. We obtained 96.88%, 95.31%, and 93.75% training, validation and testing accuracy, respectively. Figure 7 visualizes the convolutional filters learned on the NIST SD27 database by the SLN.

5.4. Matching Results

Rank identification rate provides an estimate of the probability that a matching rolled fingerprint is identified correctly at least at rank-k during a search with a latent candidate. Figure 8(a) shows the cumulative match characteristics (CMC) curve of the proposed approach in matching 258 latent fingerprints in NIST SD27 database against database of 2,257 rolled fingerprints. Figure 8(b) shows CMCs of matching the three categories of latent fingerprints in the NIST SD27 database (88 Good, 85 Bad, and 85

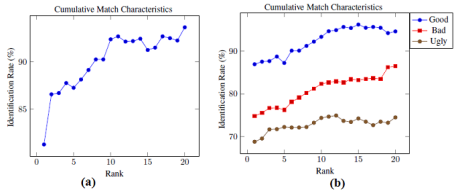

Fig. 8: (a) CMC curve of the proposed approach in matching 258 latent fingerprints in NIST SD27 database against a test database of 2,257 rolled fingerprints. (b) CMC curves for the 258 latent fingerprints in the NIST SD27 database that were grouped by subjective quality into Good (88), Bad (85), and Ugly (85).

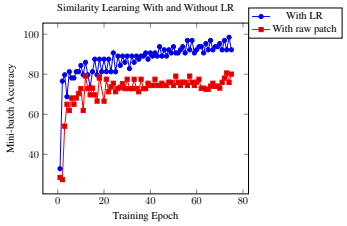

Fig. 9: Plot showing mini-batch accuracy during the training of the SLN with and without learned representation (LR) of patches. Using LR to train the SLN boosts its performance.

Ugly) [8] against the test database of 2,257 rolled prints. The plot shows the rank-k identification rate against k, $k = 1, \ldots, 20$. We obtained a rank-1 identification rate of 81.25% and a rank-20 identification rate of 93.65% on matching the 258 latent fingerprints. These results look promising when compared to a state-of-the-art rank-1 and rank-20 identification rates of 74.0% and 82.9%, respectively, reported in [9], which to the best of our knowledge, is the current state-of-the-art result. It should be noted that a test database of 29,257 fingerprints was used by the authors in [9] against 2,257 used in this work.

5.5. LR and SLN Performance

Figure 9 shows the comparison of the performance of the SLN when learned representations (LR) of patches are used for training versus when raw patches are used. The figure shows that using learned representation boosts the mini-batch accuracy of the SLN.

6. CONCLUSIONS AND FUTURE WORK

This paper proposes a patch based latent fingerprint matching system based on the similarity of learned representations encoded in a deep neural network. Our model learns patch representation and similarity function that make matching of patches invariant to rotation. We tested the proposed system by matching segmented fingerprints from 258 latent fingerprints in NIST SD27 against a database consisting of 2,257 rolled fingerprints from two different NIST databases, and achieved a rank-1 identification rate of 81.25%. This is a significant improvement on the state-of-the-art rank-1 identification rate, which to the best of our knowledge is 74%. Part of our future work is to use a deep learning approach for minutiae extrattion, as well as exploring the performance of the proposed approach on a larger fingerprint database with mixed images resolutions.

7. REFERENCES

[1] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, Aug 2007.

[2] M. A. Carreira-Perpinan and G. Hinton. On contrastive divergence learning. In *AISTATS*, volume 10, pages 33–40. Citeseer, 2005.

[3] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *J. Mach. Learn. Res.*, 11:625–660, Mar. 2010.

[4] J. Ezeobiejesi and B. Bhanu. Latent fingerprint image segmentation using deep neural network. In *Deep Learning for Biometrics*, pages 83–107, 2017.

[5] J. Feng and A. K. Jain. Filtering large fingerprint database for latent matching, Dec 2008.

[6] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3279–3286, June 2015.

[7] G. Hinton. A practical guide to training restricted boltzmann machines, version 1. 2010.

[8] M. Indovina, R. A. Hicklin, and G. I. Kiebuzinski. Elft efs: Evaluation of latent fingerprint technologies: Extended feature sets [evaluation no. 1]. NISTIR 7775, 2011.

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

[10] T.-Y. Jea and V. Govindaraju. A minutia-based partial fingerprint recognition system. *Pattern Recognition*, 38(10):1672 – 1684, 2005.

[11] J. Li, J. Feng, and C.-C. J. Kuo. Deep convolutional neural network for latent fingerprint enhancement. *Signal Processing: Image Communication*, 60:52 – 63, 2018.

[12] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999.

[13] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 519–528, June 2006.

[14] S. A. Sudiro, M. Paindavoine, and T. M. Kusuma. Simple fingerprint minutiae extraction algorithm using crossing number on valley structure. In *IEEE Workshop on Automatic Identification Advanced Technologies*, pages 41–44, June 2007.

[15] B. Yao, G. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3466–3473, June 2012.

[16] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4353–4361, June 2015.