Deep Dense Multi-level Feature for Partial High-Resolution Fingerprint Matching

Fandong Zhang Shiyuan Xin Jufu Feng*
Key Laboratory of Machine Perception (MOE), School of EECS, Peking University
{zhangfandong, xinshiyuan}@pku.edu.cn, fjf@cis.pku.edu.cn

Abstract

Fingerprint sensors on mobile devices commonly have limited area, which results in partial fingerprints. Optical sensor can capture fingerprints at very high resolution (2000ppi) with abundant details like pores, incipients, etc. It is quite crucial to develop effective partial-to-partial high-resolution fingerprint matching algorithms. Existing fingerprint matching methods are mainly minutiae-based, with fusion of different levels of features. Their accuracy degrades significantly in our application due to minutiae insufficiency and detection error. In this paper, we propose a novel representation for partial high-resolution fingerprint, named Deep Dense Multi-level feature (DDM). We train a deep convolutional neural network that can extract discriminative features inside any local fingerprint block with certain size. We find that not only minutiae but most local blocks contain sufficient features. Moreover, we analyze DDM and find that it contains multi-level information. When utilizing DDM for partial-to-partial matching, we first extract features block by block through a fully convolutional network, next match the two sets of features pairwise exhaustively, and then select the bi-directional best matches to compute matching score. Experiments indicate that our method outperforms several state-of-the-art approaches.

1. Introduction

Fingerprint authentication systems are widely used on mobile devices. Mobile fingerprint sensors are generally small limited by space and cost, which results in partial fingerprints. Capacitive fingerprint sensors commonly capture fingerprints at a resolution of around *500ppi*. Partial-to-partial fingerprint matching is challenging due to feature insufficiency. Optical fingerprint sensors can capture fingerprints with resolution as high as *2000ppi*, which contain very rich details such as pores, scars, ridge contours, incipients and so on. We expect these subtle features can provide discriminative information to cover the shortage of limited area. Moreover, with these abundant details, fingerprint authentication system is much harder to crack.

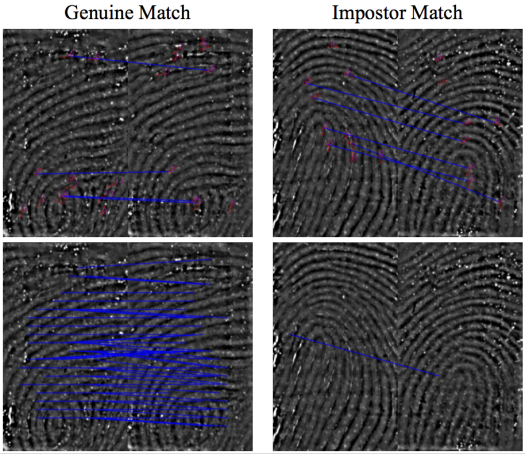

Figure 1. Compared to a minutiae-based algorithm [11] (*top row*), proposed DDM (*bottom row*) can generate denser local matches for genuine match (*left column*) of partial high-resolution fingerprint matching. Meanwhile, DDM is discriminative for impostor match with similar minutiae topologic structure (*right column*).

Existing fingerprint matching algorithms are mainly based on local matching. Generally, there are mainly two key components for local-based matching problem: where the keypoints locate and what kinds of features to extract. The most widely-used keypoint in fingerprints is minutia, which is commonly believed to be the most reliable and discriminative feature in fingerprints. However, minutiae-based algorithms have two drawbacks for our application. First, their accuracy degrades significantly when lack of minutiae. There are inadequate number of minutiae in partial fingerprints. For example, fingerprints of size $0.24" \times 0.39"$ usually contain no more than 15 minutiae. Second, minutiae detection accuracy drops when fingerprint is noisy, which also leads to bad performance. Some researchers utilize level-3 keypoints like pores, incipients and ridge contours specially for high-resolution fingerprints [1, 7, 14].

978-1-5386-1124-1/17/$31.00 ©2017 IEEE

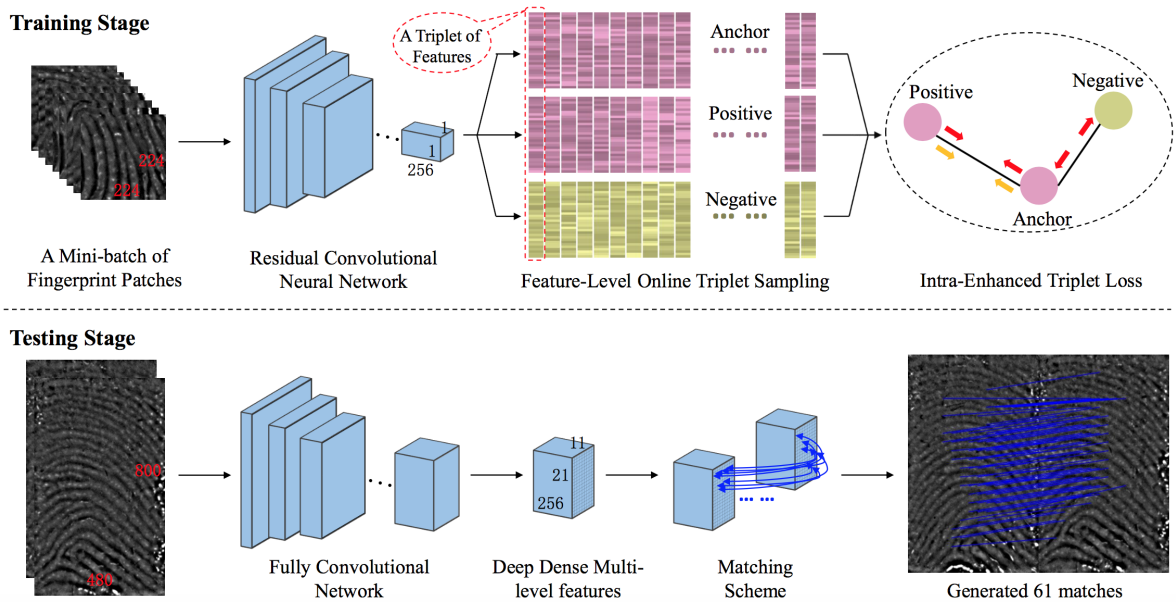

Figure 2. Proposed framework. During training stage, we take fingerprint patches as inputs. A ResNet and a metric learning method named Intra-Enhanced Triplet loss are designed to embed raw patches into an Euclidean space. We also utilize feature-level online triplet sampling to both accelerate convergence and make full use of the memory. During testing stage, we extract 11×21 of $256-dimensional$ DDMs for a partial fingerprint of size 480×800 all at a time by a fully convolutional network. When applying to fingerprint matching, we match the two feature sets point-wise, select the best matches and compute the matching score.

However, some of them don't not commonly exist for all people. For example, it has been reported that incipients are observed in only 13.5% of the fingers [22]. In fingerprints under resolution as high as *2000ppi*, level-3 features are densely distributed. Do we really need to assign specific keypoints? Here we consider extract features at any point.

Common fingerprint features include ridge flow, minutia structure, orientation field, pores, incipients, etc. These handcrafted features can not take full advantage of the high-resolution details. Deep convolutional neural network (CNN) is proved to be able to extract more effective and robust features from images. For fingerprint matching, Zhang *et al*. [31] and Lin [17] use CNN to learn global and minutia features respectively, both achieving surprising performances compared to handcrafted feature based methods. Inspired by them, we also utilize deep features.

In this paper, we propose a novel deep representation specially for partial high-resolution fingerprint, named the Deep Dense Multi-level feature (DDM). We train a deep convolutional neural network that can extract discriminative features inside any local fingerprint block with certain size. We find most high-resolution fingerprint blocks with size large enough contain sufficient discriminative information. Proposed deep network projects a fingerprint block into a vector in Euclidean space, where similarity can be measured by Cosine distance. Meanwhile, we find the learned DDMs contain multi-level information. We analyze the feature by maximum activation visualization and feature decomposition. We find that some level-3 features like pores and incipients have great responses in convolutional feature maps and the decomposed features around minutiae (level-2) and core regions (level-1) play a crucial part in the final feature.

DDM can be applied for partial high-resolution fingerprint matching. First, we extract hundreds of DDM features block by block through a fully convolutional network (FCN) [19] all at one time. Next, the extracted two feature sets are matched exhaustively. We then select the bidirectional best matches to compute matching score. Compared to a minutiae-based algorithm [11], DDM can generate denser and more discriminative local matches (Fig. 1). We evaluate the partial matching on the dataset provided by [31]. Our approach achieves an FMR10000 of 13.33, an FMR100000 of 17.11, a ZeroFMR of 20.78, and an EER of 3.35, which outperforms several state-of-the-art methods.

2. Related Work

Most existing fingerprint matching approaches are mainly based on minutiae matching, with fusion of different levels of features. Many kinds of representations are proposed for minutia matching, such as local minutia topologic structures [3, 5, 6, 11, 29], ridge patterns like orientation and frequency [9, 10, 23, 25], local texture information [27], level-3 features around minutiae [14] and so on. Recently, some researchers apply deep convolutional neural

networks to learn features around minutiae. Lin [17] utilize a ConvNet to embed enhanced and aligned local patches around minutiae into an 512-dimensional vector, while Cao *et al*. [4] train a set of ConvNets with multiple local patches to boost the accuracy for latent fingerprint matching. They both achieve promising performances.

In addition to minutiae features, some researchers introduce global features to capture the overall texture information. The most popular one remains the FingerCode [15]. After filtering the reference point region by a bank of Gabor filters, each fingerprint is represented by a 640-dimensional feature vector. Zhang *et al*. [31] utilize a ConvNet to project each partial fingerprint into an Euclidean space. However, global features lose most of the spatial information. They are generally not robust to translation. Some kinds of level-3 features are used to boost the performance. Agrawal *et al*. [1] use pores as anchor points and match the LBP histograms between them. Chen *et al*. [7] compute the matching score of extracted dots and incipients after alignment by a minutiae-based matcher. Besides, some other descriptors such as AKAZE [2], SIFT [20] can generate denser local matches between fingerprints [21, 30] than minutia, but they are not designed to make full use of fingerprint structures.

3. The Proposed Method

In this section, we first formulate the entire feature learning and matching pipeline depicted by Fig.2. Next, we describe how to collect the training patches. We then explain the training and matching procedures in detail.

3.1. Problem Formulation

During training stage, we use fingerprint patches as input. We utilize a 37-layer deep ResNet [12] with only $920K$ parameters and $140M$ FLOPS (Table 1) for feature embedding. We design a thin but deep ResNet for both computational efficiency and representation ability. We use the output of the global average pooling as feature. To constrain features in an Euclidean space, we design a metric learning function named Intra-Enhanced Triplet loss, in which intra-class similarities are enhanced compared to standard triplet loss. We also L_2 normalize the features suggested by [24]. Meanwhile, the similarity between two features can be measured by inner-product, which makes the pairwise matching simpler (see 3.4 for details).

During testing stage, we use the entire partial fingerprints as input. Instead of cropping patches and forward them one-by-one, we use an FCN [19] to extract all DDMs at one time for efficiency. For a partial fingerprint of size 480×800, we extract 11×21 DDMs (Fig. 3). Then, we match DDMs of two partial fingerprints and compute matching score.

Table 1. Detailed network architecture.

Layer Name	Output Size	Kernel
conv1	$112 \times 112 \times 16$	7×7, 16, stride 2
pool1	$56 \times 56 \times 16$	2×2, max, stride 2
conv2_x	$56 \times 56 \times 32$	$\begin{bmatrix} 1 \times 1, 16 \\ 3 \times 3, 16 \\ 1 \times 1, 32 \end{bmatrix} \times 2$
pool2	$28 \times 28 \times 32$	2×2, max, stride 2
conv3_x	$28 \times 28 \times 64$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 64 \end{bmatrix} \times 3$
pool3	$14 \times 14 \times 64$	2×2, max, stride 2
conv4_x	$14 \times 14 \times 128$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 128 \end{bmatrix} \times 4$
pool4	$7 \times 7 \times 128$	2×2, max, stride 2
conv5_x	$7 \times 7 \times 256$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
pool5	$1 \times 1 \times 256$	7×7, average, stride 7

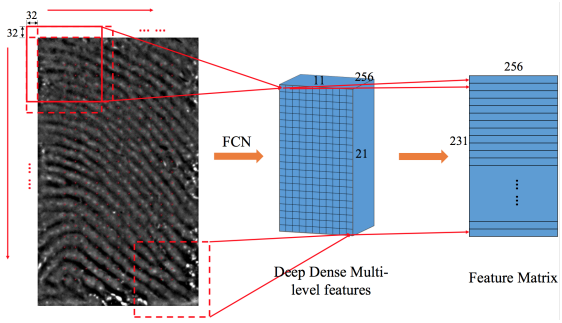

Figure 3. Procedures of extracting DDMs for partial fingerprint matching during testing stage.

3.2. Creating the Training Dataset

Generating training data is challenging since we don't know which patches correspond to the same region of the same finger. Since local areas suffer less from non-linear distortion, we assume the transformations between partial fingerprints are rigid. We design a bundled alignment method to register all fingerprints with the same identity at one time, which will be discussed in detail in Section 4.

The training dataset includes 1542 identities and 60 scans for each identity. All scans are captured from similar orientations for the simplicity of alignment. For each identity, we map all partial fingerprints into the selected absolute coordinate system after alignment. Therefore, given position of a rectangle, we can crop a patch from each partial fingerprint. We discard patches with valid pixels under 75%. Two patches overlapped under 75% belong to different classes. We finally generate 921,511 patches of 52,592

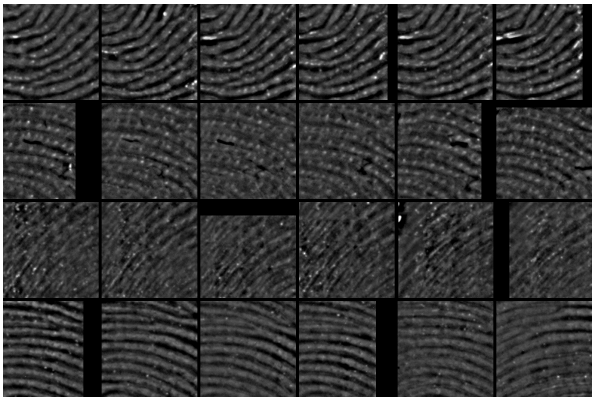

Figure 4. Examples of training patches. Samples in the same row are from the same class.

classes for training. In Fig. 4, we show some examples of training patches. Samples from the same class roughly correspond to the same fingerprint region. Our bundled alignment method achieves promising accuracy.

3.3. Intra-Enhanced Triplet Embedding

Here we describe the intra-enhanced triplet loss and the triplet sampling policy. Let $f(\cdot)$ denote the deep feature embedding.

Standard Triplet Loss is trained on triplets of images $\{(x_i^a, x_i^p, x_i^n)\}$, where x_i^a (*anchor*) and x_i^p (*positive*) have the same class labels while x_i^n (*negative*) has a different label. The loss (Eq. 1) encourages the embedding $f(\cdot)$ to ensure the distance between $f(x_i^a)$ and $f(x_i^n)$ is larger than distance between $f(x_i^a)$ and $f(x_i^p)$ plus a given margin α.

$$J_{triplet} = \frac{1}{N_t} \sum_{i}^{N_t} max\left\{0, J_i\right\}, \tag{1}$$

$$J_i = \left\|f(x_i^a) - f(x_i^p)\right\|^2 + \alpha - \left\|f(x_i^a) - f(x_i^n)\right\|^2, \tag{2}$$

where N_t is the number of triplets. Please refer to [24] for more details.

Intra-Enhanced Triplet Loss Standard triplet loss can't ensure that intra-class distances are smaller than a given threshold, which is important to matching problems. Inspired by [18], we add an *anchor-center* loss (Eq. 3) to enhance intra-class similarities.

$$J_{ac} = \frac{1}{N_a} \sum_{i}^{N_a} max\left\{0, \left\|f(x_i^a) - c^a)\right\|^2 - \beta\right\}, \tag{3}$$

where N_a is the number of *anchors*, c^a is the center point of all features of the same label with x^a within a batch, and β represents the given threshold of genuine match distances. Let λ denote the weight of J_{ac}, then the intra-enhanced triplet loss is defined as:

$$J_{iet} = J_{triplet} + \lambda J_{ac}. \tag{4}$$

In our experiments, the parameters α, β and λ are set to 0.4, 0.3, 0.5 respectively.

Feature-Level Online Triplet Sampling. The total number of triplets is quite huge. Random sampling triplets may lead to slow convergence. Therefore, we adopt online hard triplet mining as in [24], which can significantly accelerate the convergence. Each mini-batch is composed of two sets of instances, *anchor-positive* set and *negative* set. We first sample n^{cp} categories and put all instances belong to them into *anchor-positive* set, and then select n^{cn} instances from other categories (one for each category) to form the *negative set*. To make full use of the memory, instead of using a three-branch siamese architecture, we forward all images in a mini-batch with one branch and sample the triplets on feature level. For each *anchor-positive* feature pair, we select the nearest *negative* feature to the *anchor* to form a triplet. Therefore there will be $O(N_b^2)$ of triplets sampled from a batch of N_b instances. In our experiments, we set n^{cp}, n^{cn} and N_b as 20, 76, 256 respectively.

3.4. Matching Scheme

For each fingerprint i, let $\{f_i^1, f_i^2, \cdots, f_i^m\}$ denote the feature set, where m is the number of features. Let $\mathbf{F}_i \in \mathbb{R}^{m \times d}$ denote the feature matrix (Fig. 3), where the jth row is f_i^j and d is the feature dimension. Define $\mathbf{M}_{ij} = \mathbf{F}_i \mathbf{F}_j^T$ as the matching matrix, and let M_{ij}^{uv} denote the (u, v)th entry of $\mathbf{M}_{ij}$. Since we normalize features, M_{ij}^{uv} is the matching score of (f_i^u, f_j^v). We discard all matches with score under $1 - \beta$, and select the bi-directional best matches, i.e. M_{ij}^{uv} that is the maximum element both in the uth row and the vth column. We simply sum up the scores of selected matches as the matching score of two partial fingerprints.

4. Bundled Fingerprints Alignment

In this section, we introduce a method to align a set of fingerprints with the same identity. Actually, it's not necessary to align all samples for training. We just need to make sure most fingerprints are aligned correctly and the false ones can be figured out and discarded. Simple pairwise and growth alignment will lead to accumulated errors. Inspired by [26], we minimize the global projection error to find transform parameters of all partial fingerprints at the same time. Specifically, as mentioned above, we collect training fingerprints with roughly the same orientation. Therefore, we only need to solve the translation parameters. Meanwhile, an absolute coordinate system is built. Figure 5 shows the pipeline of the proposed alignment. Following are the key steps:

4.1. Pair-wise Fingerprint Matching

Each fingerprint is matched to the others using [11], getting a minutiae-based matching score $Score_{mnt}(i, j)$. This method uses only minutia topological information, which

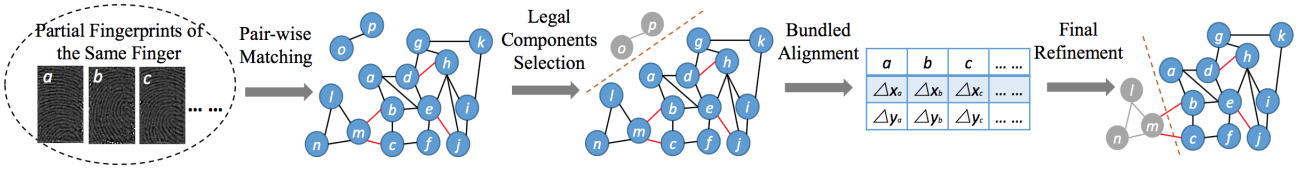

Figure 5. Pipeline of bundled fingerprints alignment. After pair-wise matching, there are still some false matches (shown as *red* edges). Legal components selection can discard small isolated connected nodes *(o, p)*. However, the edges between nodes *(l-n)* and nodes *(a-k)* are both incorrect, which means there are no correct constraints between the two nodes groups. After final refinement, *(l-n)* are discarded.

brings lots of false matches. We then check the orientation field similarity (Eq. 5). We only select the fingerprint pairs with both $Score_{mnt}(i,j)$ and $Score_{of}(i,j)$ above given thresholds.

$$Score_{of}(i,j) = 1 - \frac{1}{p_{ij}} \sum_{k}^{p_{ij}} \left| sin(o_i^k - o_j^k) \right| \tag{5}$$

where o_i^k and o_j^k denote intersected orientation field of two fingerprints, and p_{ij} denotes the size of the intersection.

4.2. Legal Components Selection

Not all fingerprints can be aligned, e.g. fingerprint o and p in Fig. 5. Let $\mathcal{M}$ denote the set of all valid fingerprint matching pairs. We establish a fingerprint association graph, denoted as G, in which nodes represent fingerprints and the two nodes are connected if the corresponding fingerprint-pair is in $\mathcal{M}$. Fingerprints from different connected sub-graphs of G cannot be aligned since there are no constraints between them. Therefore, we find the largest connected set G_l and discard other fingerprints. Let $\mathcal{M}_l$ denotes the set of remaining fingerprint-pairs, and l denotes the number of fingerprints in G_l. For simplicity, we assume indices of fingerprints in G_l are $\{1, 2, \cdots, l\}$.

4.3. Bundled Alignment

Let $\mathbf{t}_i = [\Delta x_i, \Delta y_i]^T$ denote the translation of the ith fingerprint, and $\mathbf{T}$ denote the translation matrix:

$$\mathbf{T} = [\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, \cdots, \Delta x_l, \Delta y_l]^T_{2l \times 1}. \tag{6}$$

The global projection error is defined as:

$$J(\mathbf{T}) = \sum_{(i,j) \in \mathcal{M}_l} \frac{1}{m_{ij}} \sum_{k}^{m_{ij}} \left\| (\mathbf{x}_{ij}^k + \mathbf{t}_i) - (\mathbf{x}_{ji}^k + \mathbf{t}_j) \right\|^2, \tag{7}$$

where m_{ij} is the number of minutia-matches of the fingerprint-pair (i,j), and $(\mathbf{x}_{ij}^k, \mathbf{x}_{ji}^k)$ is the coordinates of the kth minutia-match of the fingerprint-pair (i,j). To solve the transform parameters, we lift $\mathbf{T}$ by a set of selecting matrices $\{\mathbf{C}_i | i = 1, 2, \cdots, l\}$, where $\mathbf{C}_i$ is a $2 \times 2l$ matrix of zeros with $\mathbf{I}_{2\times 2}$ in place of the ith 2×2 block,

$$\mathbf{C}_i = \begin{bmatrix} \mathbf{0}^1_{2\times 2} & \cdots & \mathbf{I}^i_{2\times 2} & \cdots & \mathbf{0}^l_{2\times 2} \end{bmatrix}_{2 \times 2l} \tag{8}$$

Therefore,

$$\mathbf{t}_i = \mathbf{C}_i \mathbf{T} \tag{9}$$

$$J(\mathbf{T}) = \mathbf{T}^T \mathbf{A} \mathbf{T} + 2\mathbf{B}\mathbf{T} + \mathbf{D}, \tag{10}$$

$$\mathbf{A} = \sum_{(i,j) \in \mathcal{M}_l} (\mathbf{C}_i - \mathbf{C}_j)^T (\mathbf{C}_i - \mathbf{C}_j), \tag{11}$$

$$\mathbf{B} = \sum_{(i,j) \in \mathcal{M}_l} (\frac{\sum_k^{m_{ij}} (\mathbf{x}_{ij}^k - \mathbf{x}_{ji}^k)^T}{m_{ij}}) (\mathbf{C}_i - \mathbf{C}_j), \tag{12}$$

$$\mathbf{D} = \sum_{(i,j) \in \mathcal{M}_l} \frac{1}{m_{ij}} \sum_{k}^{m_{ij}} \left\| \mathbf{x}_{ij}^k - \mathbf{x}_{ji}^k \right\|^2. \tag{13}$$

We select the first fingerprint as the global coordinate system ($\mathbf{t}_1 = \mathbf{0}_{2\times 1}$). Then, we separate the $\mathbf{t}_1$ from $\mathbf{T}$:

$$\mathbf{T} = [\mathbf{t}_1^T, \mathbf{T}_{\setminus 1}^T]^T \tag{14}$$

Therefore, Eq.(10) can be written as,

$$J(\mathbf{T}_{\setminus 1}) = \mathbf{T}_{\setminus 1}^T \mathbf{A}_{\setminus 1 \setminus 1} \mathbf{T}_{\setminus 1} + 2\mathbf{B}_{\setminus 1} \mathbf{T}_{\setminus 1} + \mathbf{D}, \tag{15}$$

$$\mathbf{A}_{\setminus 1 \setminus 1} = \mathbf{A}_{3:2l,3:2l}, \mathbf{B}_{\setminus 1} = \mathbf{B}_{:,3:2l}, \tag{16}$$

where ':' denotes all elements within the given index. It's simple to prove that $\mathbf{A}_{\setminus 1 \setminus 1}$ is full-rank. Therefore, the optimal transform parameter matrix $\mathbf{T}^*_{\setminus 1}$ is unique:

$$\mathbf{T}^*_{\setminus 1} = -\mathbf{A}^{-1}_{\setminus 1 \setminus 1} \mathbf{B}^T_{\setminus 1} \tag{17}$$

4.4. Final Refinement

After the first step in Section 4.1, there are still a few false matches. If the set of false matches corresponds to an edge cut set of G_l, the bundled alignment will fail. For example, in Fig. 5, edges *bm* and *cm* are both incorrect, which means there are no correct constraints between the two nodes groups *(a-k)* and *(l-n)*. To address this problem, we delete the mis-aligned fingerprints as follows. Define two fingerprints are incompatible if their orientation field similarity (Eq.(5)) is two small. We first transform all the fingerprints to the global coordinate system. Then we keep deleting the fingerprint with the maximum incompatibility with others until all fingerprints are incompatible.

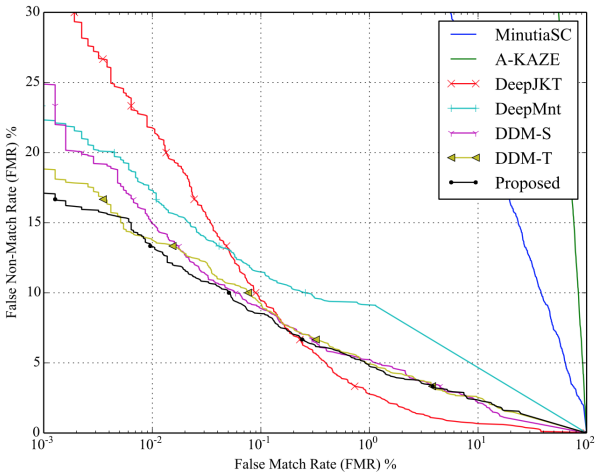

Figure 6. ROC curves for several state-of-the-art methods and proposed DDM-based approaches.

5. Experiments

5.1. Dataset

We evaluate the performances on the dataset provided by [31], in which images are 480×800 and captured by an optical fingerprint sensor with area $0.24" \times 0.39"$. There are 180 different fingers from 30 people. For each identity, we pick up 20 enrolled templates and 10 testing scans with random regions and similar orientations for verification. Here we have 1800 genuine matches and 312000 impostor matches.

5.2. Implementation Details

The deep networks in our experiments are implemented with Caffe [16]. We adopt settings like batch normalization [13], weights initialization, stochastic gradient descent policy from [12]. We train all networks from scratch for 3×10^5 iterations without any pre-training. We randomly rotate each image between -20 and 20 degrees and crop a region of 224×224 pixels to augment the training data.

5.3. Baseline Methods

There are only a few algorithms proposed to address the partial high-resolution fingerprint matching problem and none of them opens the source codes. We establish four baseline methods as follows:

MinutiaSC is a minutiae-based algorithm adopted from [11]. We resize all images into 500dpi as required, enhance the ridges using STFT analysis [8], next detect minutiae using [28], and then use source code of [11] for global matching and matching score computing.

A-KAZE is an algorithm based on A-KAZE feature adopted from [21]. We resize all images into 500dpi respectively. We implement feature extraction and matching with OpenCV 3.1.0.

DeepJKT is adopted from [31]. It embeds each partial fingerprint into a 512-dimensional feature vector in Euclidean space with a deep ResNet. We re-implement the method with our own training data.

DeepMnt is a variation of [17], which utilizes a ConvNet to learn features around minutiae. We train the same 37-layer network with intra-enhanced triplet loss, and then use [11] for global matching.

We also compare with two variants to reveal the contribution of intra-enhanced triplet loss by replacing the loss to softmax with cross-entropy loss (denoted as **DDM-S**) and standard triplet loss (denoted as **DDM-T**) respectively.

Table 2. Matching evaluations (%).

Method	EER	FMR 1000	FMR 10000	FMR 100000	Zero FMR
MinutiaSC	17.78	58.49	71.17	82.56	83.89
A-KAZE	41.16	88.89	93.17	95.83	98.50
DeepJKT	**1.87**	9.50	21.78	34.11	41.44
DeepMnt	9.06	11.50	17.33	22.33	35.56
DDM-S	3.50	8.94	14.94	24.89	33.06
DDM-T	3.45	9.28	13.78	18.83	23.72
Proposed	3.35	**8.56**	**13.33**	**17.11**	**20.78**

5.4. Evaluations

We evaluate the matching results by equal error rate (EER), false non-match rate (FNMR) with false match rate (FMR) under $1/1000$ (FMR1000), FMR10000, FMR100000 and ZeroFMR. We add FMR10000 and FMR100000 specifically since they are very important to mobile fingerprint authentication in practice.

Table 2 shows the matching evaluations. Compared to the above baseline models, proposed approach achieves top performances for almost all protocols. The evaluation results can be summarized as follows:

- Compared to handcrafted features such as minutia topological structure and A-KAZE feature, deep learned features fit better for our problem for three reasons. First, handcrafted features can't make full use of the detail information. Second, limited minutia number results in similar minutia topological structures for different fingerprints as shown in Fig. 7. Third, some testing samples from [31] are very noisy, which is challenging for handcrafted features.

- DDM-based approaches outperform other deep-based approaches. DeepJKT learns global feature, which is not robust to translation. Meanwhile, global learned feature can't make good use of the subtle details, that's why the FNMR increase so fast when FMR decreases.

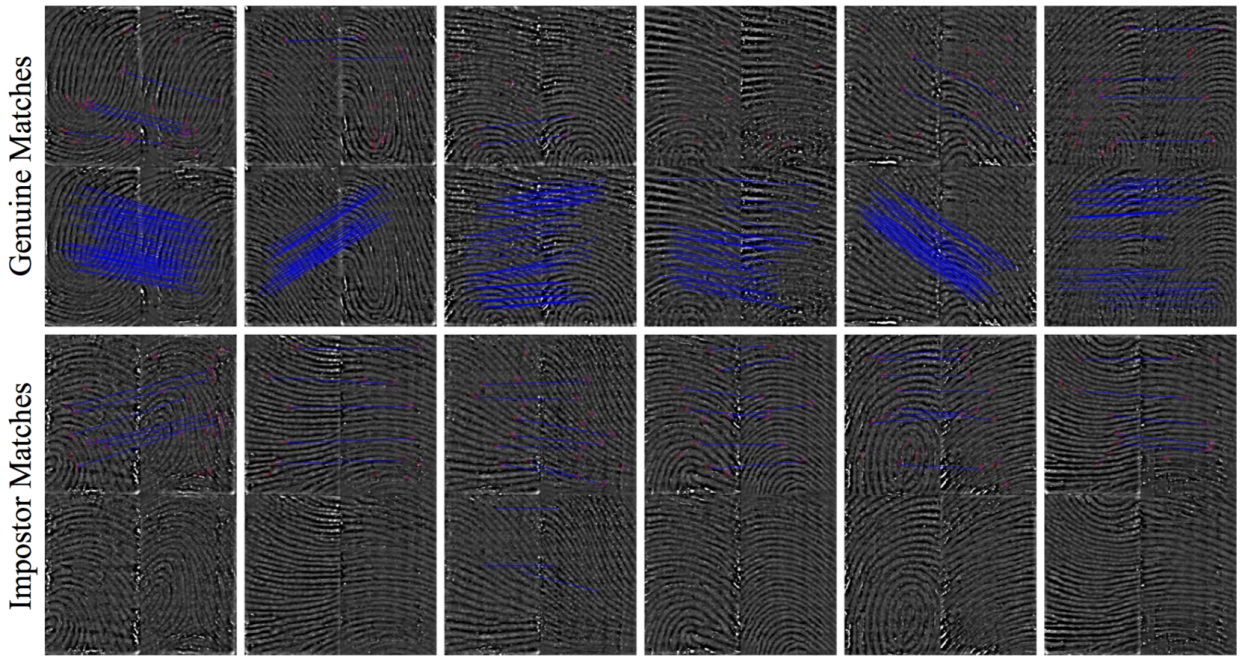

Figure 7. Examples of matching results and comparison of a minutia-based algorithm [11] and proposed approach. The two groups show genuine and impostor matches respectively. For either group, the two rows denote results of [11] and proposed approach respectively.

DeepMnt suffers from lack of reliable minutiae, which also results in very high EER.

- Intra-enhanced triplet loss can boost the performances. Softmax loss can't constrain features from the same class to be close in Euclidean space. Intra-enhanced triplet loss is slightly better than standard triplet loss due to the enhancement of intra-class similarities.

We also evaluate the processing time on Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.1GHz. Time taken for feature extraction of a fingerprint of size 800×480 and 1:1 matching is 466 ms and 6 ms respectively.

5.5. Feature Analysis

Maximum Convolutional Activation Map (MCAM) We visualize the convolutional activations to reveal the contribution of level-3 features. Given a convolution layer, let $f_k(x, y)$ represent the activation of the kth unit at spatial location (x, y). We reduce the feature maps to one heat map by max-pooling across the channel dimension. Let M_{mca} denote the MCAM, the element at each location is:

$$M_{mca}(x, y) = max\left\{f_k(x, y)\right\}. \tag{18}$$

We visualize the MCAMs of $Conv_1$ and $Conv_{2\_3}$ which are right ahead of max-pooling layers.

Feature Decomposition Map (FDM) Inspired by [32], we decompose the learned features to find out the key regions. Given an image patch, let $\mathbf{g}(x, y)$ denote the feature vector of the last convolutional layer, i.e. $Conv_{5\_3}$, at spatial location (x, y), and $\mathbf{p}$ denote the output vector of $pool5$. Thus we have,

$$\mathbf{p} = \frac{\sum_y^h \sum_x^w \mathbf{g}(x, y)}{h \times w} = \sum_y^h \sum_x^w \mathbf{p}_d(x, y), \tag{19}$$

$$\mathbf{p}_d(x, y) = \frac{\mathbf{g}(x, y)}{h \times w}, \tag{20}$$

where $h \times w$ is the feature map resolution of the $Conv_{5\_3}$, i.e. 7×7. Therefore, we decompose $\mathbf{p}$ to $h \times w$ sub-feature-vectors $\mathbf{p}_d(x, y)$. By projecting $\mathbf{p}_d(x, y)$ to the direction of $\mathbf{p}$, we obtain,

$$\mathbf{p} = \sum_y^h \sum_x^w \frac{< \mathbf{p}_d(x, y), \mathbf{p} >}{\|\mathbf{p}\|} \cdot \frac{\mathbf{p}}{\|\mathbf{p}\|} \tag{21}$$

The length of each sub-vector projected in the direction of $\mathbf{p}$ can be regarded as its contribution to the final feature. Therefore, we define M_{fd} as the FDM, where each spatial element is computed by,

$$M_{fd}(x, y) = \frac{< \mathbf{p}_d(x, y), \mathbf{p} >}{\|\mathbf{p}\|}. \tag{22}$$

Finally, we normalize M_{fd} to $[0, 1]$ since we mainly focus on the relative measurement between different sub-regions.

Multi-level Information In Fig. 8, we visualize the MCAMs and FDMs of some examples. We compare two

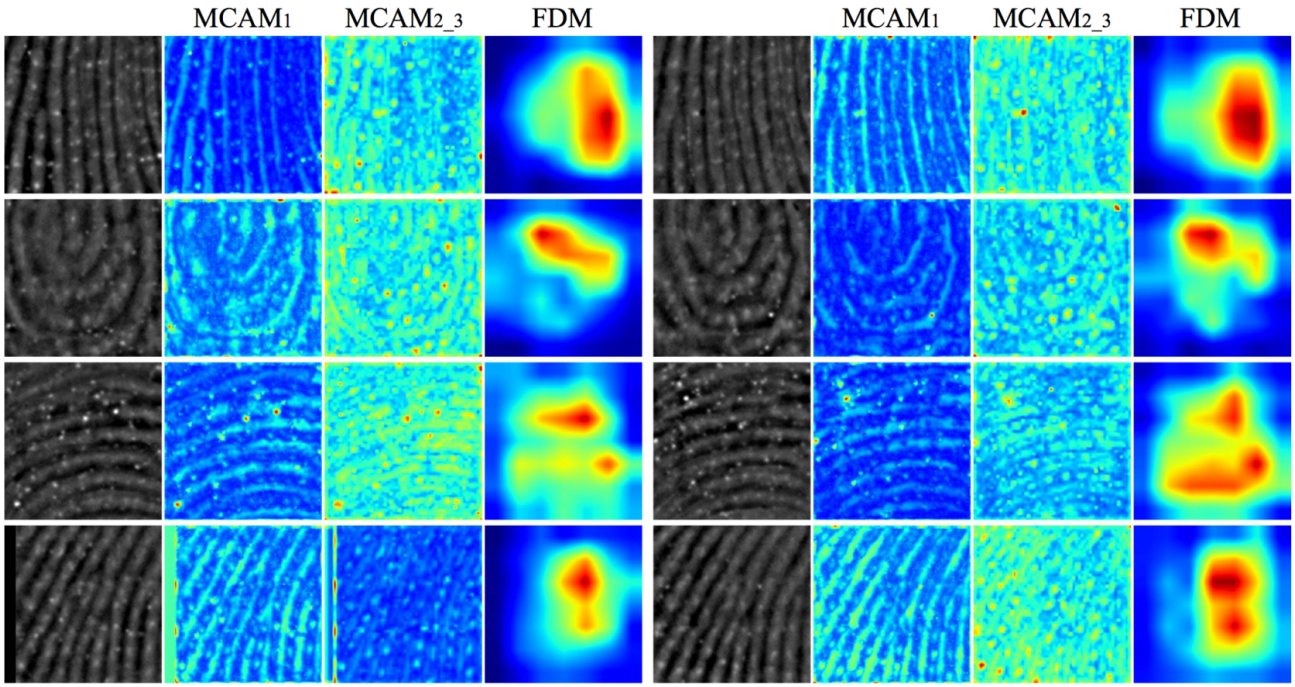

Figure 8. Visual analysis of DDMs of some examples. Each row contains two instances from the same class. For each instance, we visualize the MCAMs and FDM, where MCAM_1 and MCAM_{2_3} denote the MCAM of $Conv_1$ and Cov_{2_3} respectively.

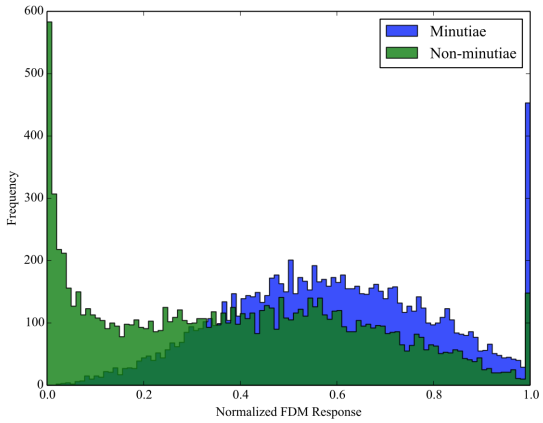

Figure 9. Histograms of FDM responses of 10000 random minutiae regions and 10000 non-minutiae regions.

instances from the same class which are shown in the same row. In MCAMs, we can see that some level-3 features like pores, edge contours are captured by convolutional layers. Activations of pores remain relatively large in MCAMs of different layers, while valley contours are enhanced especially in MCAM_1.

FDMs of both instances in each row have similar highlighted regions, which indicates the robustness and effectiveness of DDM. In the top row, the region with the highest ridge frequency responds the largest for both instances, which means that the high-frequency is probably one of the key features of this class. The second row shows both of the core regions are captured by DDM. In the bottom two rows, the highlighted regions roughly correspond to the minutiae regions. Furthermore, we randomly sample 10000 minutiae regions and 10000 non-minutiae regions, and plot the histograms of FDM responses of the two sets in Fig. 9. Responses of minutiae regions are relatively larger, which indicates DDM captures information of minutiae.

To sum up, the MCAMs and FDMs provide reasonable evidences that the learned feature DDM takes advantages of multi-level fingerprint information.

6. Conclusions and Future Work

In this paper, we propose a novel representation for high-resolution fingerprint local patches, namely Deep Dense Multi-level feature (DDM). DDM can improve the performance for partial-to-partial high-resolution fingerprint matching. In future, we will explore how to fuse minutiae topological structure information into DDM, which may be helpful to reduce the false matches.

Acknowledgments

This work was supported by NSFC(61333015).

References

[1] P. Agrawal, R. Kapoor, and S. Agrawal. Partial fingerprint matching: Fusion of level 2 and level 3 features. In *Confluence the Next Generation Information Technology Summit*, pages 504–508, 2014.

[2] P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *British Machine Vision Conference*, pages 13.1–13.11, 2013.

[3] J. Cao and J. Feng. A robust fingerprint matching algorithm based on compatibility of star structures. *Proceedings of SPIE*, 7498:74983X–74983X–7, 2009.

[4] K. Cao and A. K. Jain. Automated latent fingerprint recognition. *eprint arXiv:1704.01925*, 2017.

[5] 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.

[6] X. Chen, J. Tian, and X. Yang. A matching algorithm based on local topologic structure. In *Image Analysis and Recognition: International Conference, Iciar 2004, Porto, Portugal, September 29-October 1, 2004, Proceedings*, pages 360–367, 2004.

[7] Y. Chen and A. K. Jain. Dots and incipients: extended features for partial fingerprint matching. In *Biometrics Symposium, 2007*, pages 1–6. IEEE, 2007.

[8] S. Chikkerur, A. N. Cartwright, and V. Govindaraju. Fingerprint enhancement using stft analysis. *Pattern Recognition*, 40(1):198–211, 2007.

[9] H. Choi, K. Choi, and J. Kim. Fingerprint matching incorporating ridge features with minutiae. *IEEE Transactions on Information Forensics and Security*, 6(2):338–345, 2011.

[10] J. Feng. Combining minutiae descriptors for fingerprint matching. *Pattern Recognition*, 41(1):342–352, 2008.

[11] X. Fu, C. Liu, J. Bian, and J. Feng. Spectral correspondence method for fingerprint minutia matching. In *ICPR*, pages 1743–1746. IEEE, 2012.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[13] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *IEEE International Conference on Machine Learning*, pages 448–456, 2015.

[14] A. K. Jain, Y. Chen, and M. Demirkus. Pores and ridges: high-resolution fingerprint matching using level 3 features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):15–27, 2007.

[15] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti. Filterbank-based fingerprint matching. *IEEE Transactions on Image Processing*, 9(5):846–859, 2000.

[16] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM International Conference on Multimedia*, pages 675–678, 2014.

[17] J. Lin. A local minutiae feature matching method based on deep convolutional neural network. *Master Thesis, Peking University, China*, 2016.

[18] H. Liu, Y. Tian, Y. Wang, L. Pang, and T. Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2167–2175, 2016.

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017.

[20] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[21] S. Mathur, A. Vjay, J. Shah, S. Das, and A. Malla. Methodology for partial fingerprint enrollment and authentication on mobile devices. In *ICB*, pages 1–8, June 2016.

[22] H. Nieuwendijk. Fingerprints. *http://www.xs4all.nl/dacty/minu.htm*, 2004.

[23] J. Qi, M. Xie, and W. Wang. A novel fingerprint matching method using a curvature-based minutia specifier. In *IEEE International Conference on Image Processing*, pages 1488–1491, 2008.

[24] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015.

[25] M. Tico and P. Kuosmanen. Fingerprint matching using an orientation-based minutia descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):1009–1014, 2003.

[26] B. Triggs, P. F. Mclauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment a modern synthesis. *Lecture Notes in Computer Science*, 1883(1883):298–372, 2000.

[27] X. Wang, J. Li, and Y. Niu. Fingerprint matching using orientationcodes and polylines. *Pattern Recognition*, 40(11):3164–3177, 2007.

[28] C. I. Watson, M. D. Garris, E. Tabassi, C. L. Wilson, R. M. McCabe, S. Janet, and K. Ko. User's guide to export controlled distribution of nist biometric image software (nbis). In *Gaithersburg, MD*, 2001.

[29] W. Xu, X. Chen, and J. Feng. A robust fingerprint matching approach: Growing and fusing of local structures. In *Advances in Biometrics, International Conference, ICB 2007, Seoul, Korea, August 27-29, 2007, Proceedings*, pages 134–143, 2007.

[30] M. Yamazaki, D. Li, T. Isshiki, and H. Kunieda. Sift-based algorithm for fingerprint authentication on smartphone. In *6th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES), 2015*, pages 1–5. IEEE, 2015.

[31] F. Zhang and J. Feng. High-resolution mobile fingerprint matching via deep joint knn-triplet embedding. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 5019–5020. AAAI Press, 2017.

[32] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.