Patch-based Segmentation of Latent Fingerprint Images Using Convolutional Neural Network

Asif Iqbal Khan & Mohd Arif Wani

APPLIED ARTIFICIAL INTELLIGENCE
2019, VOL. 33, NO. 1, 87–100
https://doi.org/10.1080/08839514.2018.1526704

Patch-based Segmentation of Latent Fingerprint Images Using Convolutional Neural Network

Asif Iqbal Khan and Mohd Arif Wani

Department of Computer Science, University of Kashmir, Srinagar, India

ABSTRACT
Latent fingerprint segmentation involves marking out all the foreground regions accurately in a latent fingerprint image, but due to poor quality images and complex background, segmentation of latent fingerprint images is one of the most difficult tasks in automatic latent fingerprint recognition systems. In this article, we propose a patch-based technique for segmentation of latent fingerprint images, which uses Convolutional Neural Network (CNN) to classify patches. CNN has recently shown impressive performance in the field of pattern recognition, classification, and object detection, which inspired us to use CNN for this complex task. We trained the CNN model using SGD to classify image patches into fingerprint and non-fingerprint classes followed by proposed false patch removal technique, which uses "majority of neighbors" to remove the isolated and miss-classified patches. Finally, based on the final class of patches, an ROI is constructed to mark out the foreground from the background of latent fingerprint images. We tested our model on IIIT-D latent fingerprint database and the experimental results show improvements in the overall accuracy compared to existing methods.

Introduction

Latent fingerprints are the hidden fingerprints that we unintentionally leave on the surface of objects. Latent fingerprints left at crime scenes are useful evidence in the court of law and crime investigation departments have been using latent fingerprints for more than 100 years now (Sankaran, Vatsa, and Singh 2014). Latent fingerprints are usually not directly visible to naked eyes and can be lifted or photographed using special chemicals and procedure in order to be used as evidence in court proceedings.

Latent fingerprints can be present on any surface like glass, cup, newspaper, table, etc. and very often these surfaces are not clear or regular, thus making it difficult to extract fingerprint from these surfaces. Unlike normal rolled fingerprint images, latent fingerprint images have very poor quality and complex background.

CONTACT Asif Iqbal Khan khanasifiqbal7@gmail.com Department of Computer Science, University of Kashmir, J&K, India – 190006.

Their ridge structure is not clean and contains stains, spikes, lines, text, etc. thus making the segmenting of foreground regions very difficult. Figure 1 below shows some sample latent fingerprint images from IIIT-D latent fingerprint database. Segmentation of latent fingerprint is the first step in latent fingerprint processing. Latent fingerprint experts manually mark the region-of-interest in latent fingerprints. The segmented fingerprint is then enhanced followed by feature extraction and matching (Sankaran, Vatsa, and Singh 2014) (Ezeobiejesi and Bhanu 2016). In order to fully automate the process of latent fingerprint recognition, automatic segmentation of latent fingerprint is need of the hour. Latent fingerprint segmentation is a challenging task and thus very few researchers have worked on latent fingerprint segmentation (Sankaran, Vatsa, and Singh 2014).

In recent years, deep learning techniques have achieved great success and won numerous contests in pattern recognition and machine learning (Schmidhuber 2015). Deep neural network techniques have shown tremendous performance in computer vision and pattern recognition. One of the deep learning technique called as Convolutional Neural Network (CNN) is a neural network architecture with multiple hidden layers, which uses local connections known as *local receptive field* and *weight-sharing* for better performance and efficiency. The deep architecture helps these networks learn many different and complex features which a simple neural network cannot learn. CNN-based techniques (Wan et al. 2014) have evolved as powerful visual models and achieved state-of-the-art performance in solving different problems of computer vision and pattern recognition like object detection and image classification. Recently published research papers have shown that CNNs can also deliver outstanding performance on more challenging visual classification tasks like semantic segmentation, etc. (Szegedy, Toshev, and Erhan 2013).

Literature Review

Fingerprint recognition, which has been one of the widely discussed topics in the field of computer vision, still faces many challenges. Latent fingerprint segmentation, fingerprint liveness detection, and low quality fingerprint

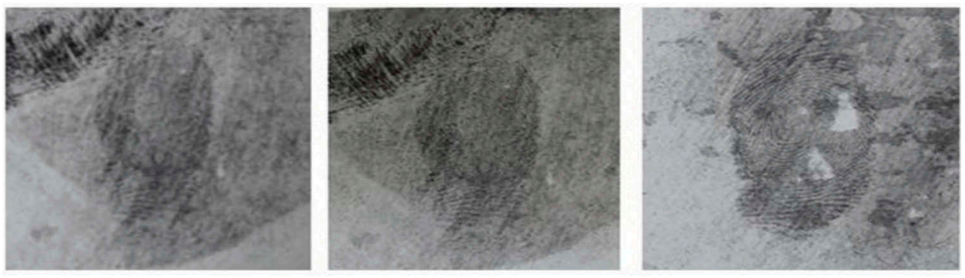

Figure 1. Sample latent fingerprint images from IIIT-D database.

images are some of the existing challenges in the field of automatic fingerprint recognition (Khan and Wani 2014, 2015). The early work on automatic segmentation of latent fingerprints was done by Karimi and Kuo (Karimi-Ashtiani and Jay Kuo 2008) in 2008. They used local windows to compute frequency and orientation components followed by calculation of inter-ridge distances for reliability. Nathan Short (Short et al. 2011) proposed a technique which uses ridge template correlation. A latent fingerprint is cross-correlated with fingerprint template of ridge pattern. The cross-correlation is done region-wise and then these regions are classified as foreground and background based on the result of cross-correlation. This technique achieved an Equal Error Rate (EER) of 33.8% on NIST SD-27 database. H. Choi et.al (Choi et al. 2012), used orientation and frequency of local regions and removed the structured noise in background. They used Fourier analysis to estimate the local frequency in the latent fingerprint image and valid frequency regions are used to identify fingerprint region. Zhang (Zhang, Lai, and Jay Kuo 2012) proposed an adaptive total variation (TV) model for latent fingerprint segmentation. They identified different background noise patterns like arches, stains, characters, lines, etc. The adaptive model then dynamically adjusts the fidelity coefficient that separates the identified background patterns from foreground. Arshad et al used a clustering technique to divide latent fingerprint image into blocks and then the standard deviation for each block is calculated. If the standard deviation of a block is less than a predefined threshold then it is marked as background block otherwise, marked as foreground block (Arshad, Raja, and Khan 2014). Inspired by the success of CNNs in various fields like pattern recognition, object detection, and classification, Cao and Jain (Cao et al. 2015) used CNN for orientation field estimation in latent fingerprints. They posed latent orientation field estimation in a latent fingerprint to a classification problem, and proposed a CNN-based approach for estimation of orientation field. Jude Ezeobiejesi and Bir Bhanu proposed a latent fingerprint segmentation algorithm based on fractal dimension features and weighted extreme learning machine. The algorithm partitions a latent fingerprint image into 8×8 patches and then uses a weighted extreme learning machine classifier to classify the patches into fingerprint and non-fingerprint classes (Ezeobiejesi and Bhanu 2016). Jonathan Long (Long, Shelhamer, and Darrell 2015) fine-tuned existing classification CNN models like ALexNet and VGG net and transformed them into fully connected CNN to perform segmentation of images.

Convolutional Neural Network

CNNs are neural network architectures that use three ideas: local receptive field, extensive weight-sharing, and pooling. The area (local region) where the filter is applied is called as the local receptive field. CNN consists of a sequence of different types of layers which include convolutional layer, pooling layer, and fully connected layer. These layers are stacked up to form a full CNN architecture.

Convolutional Layer

Convolution layer is the core building block of CNN, which involves shift, multiply, and sum operations. Its parameters consist of a set of learnable filters or weight matrices also known as kernels. The aim of the convolutional layer is to extract patterns found within local regions of the input images that are common throughout the dataset. The convolution operation involves element wise multiplication of values in receptive field with the filter. The weighted sum is then added with a bias and passed through an activation function to introduce non-linearity. The filter then moves forward by a number of steps specified by a parameter called as stride. Same operation is repeated to obtain the next value using the same filter. This way a feature map is obtained for each filter in each convolutional layer. The convolution operation for five local connection networks and one dimensional data can be expressed as:

$$v_j = \sum_{i=1}^{l} w_i x_{i+j-1} \quad \mathrm{j} = 1,\ 2,\ 3,\ 4,\ 5. \tag{3.1}$$

where w is the weight matrix shared by all hidden neurons and l is the kernel size. Equation 3.1 above is in the form of convolution sum and that is the reason a network described in this manner is referred as convolutional neural network (Krizhevsky, Sutskever, and Hinton 2012). Feature map is obtained after adding a bias term and then applying a non-linear function to Equation 3.1 above and can be expressed as follows:

$$h_k = \sigma\left(v_j^k + b_k\right)$$

where $b_{k\ is}$ the bias term, h_k is the k-th feature map and σ is the non-linearity function.

Kernel or Filter: The weights in each convolutional layer specify the convolution filters and there may be multiple filters in each convolutional layer. Every filter that contains some feature is a small portion of an image and during forward pass each filter is slid or convolved across the dimensions of the input image producing feature map of that filter. The output is a measure of how well the kernel matches each local receptive field of the image and is briefly explained by Figure 2. The figure shows an input image and a kernel which is convolved over the input image to get a feature map. The feature map obtained shows how well the kernel (T shaped pattern in this case) matches the input image.

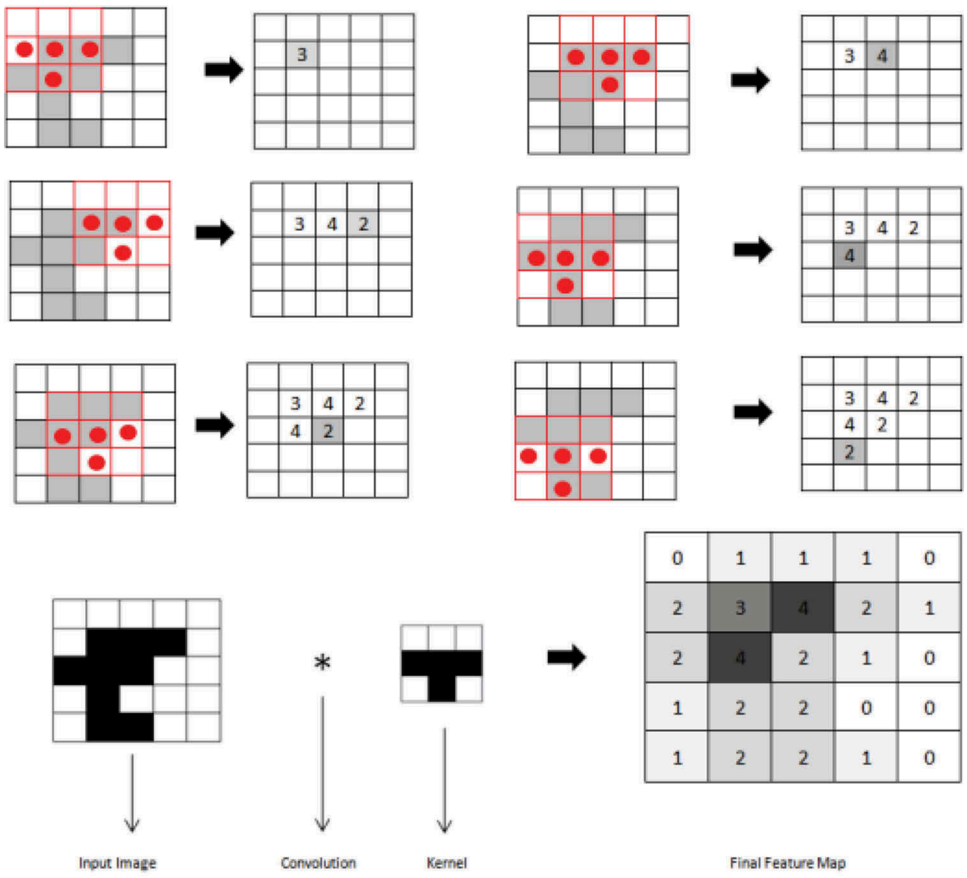

Figure 2. Example of convolution operation.

Pooling Layer

Each convolution layer is followed by an optional pooling layer to simplify the information in the output from the convolutional layer. A pooling layer takes each feature map output from the convolutional layer and down samples it, i.e., pooling layer summarizes a region of neurons in the convolution layer. The main purpose of the pooling layer is to reduce the spatial size of the input thus reducing the parameters and computation in the network. There are many pooling techniques available and some of the mostly used pooling techniques are Max Pooling and Mean Pooling.

Max pooling simply outputs the maximum value in the input region. Input region is a variable size subset of the input data. For example, if input region is of size 2 × 2 the max-pooling unit will output the maximum of the four values as shown in Figure 3. Mean Pooling outputs the mean of the all the values in the input region.

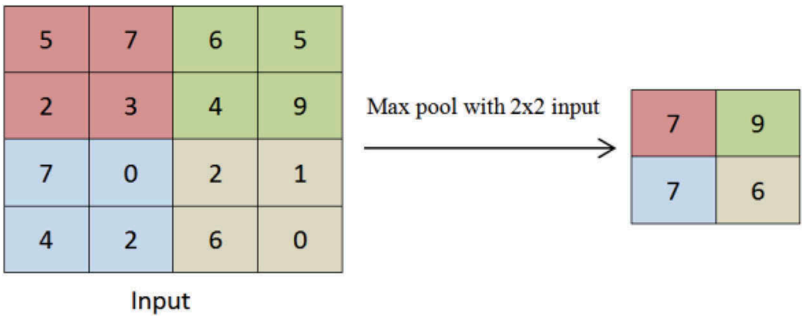

Figure 3. Max pooling.

Fully Connected Layer

In a typical CNN architecture, convolutional layers are followed by fully connected layer. In fully connected layer each neuron from previous layer is connected to every neuron in the next layer and every value gets a vote for predicting how strongly a value matches a particular class.

Proposed CNN Model for Segmentation

Algorithm Overview

The proposed segmentation technique divides the input image into 16 × 16 blocks/patches and then uses a Convolutional Neural Network (CNN) model (called Patch Classifier) to classify them into fingerprint and non-fingerprint blocks. False blocks eradication technique is used to dismiss misclassified and isolated blocks. The blocks are finally assembled back to form a segmented image. The block diagram of the proposed model is shown in Figure 4. Figure 5 shows the architecture of the proposed CNN model. The CNN consists of 3 convolutional layers with weights, one subsampling layer, 2 fully connected layers and one dropout layer. The output of the last fully-connected layer is fed to a 2-way softmax classifier which produces a distribution over the 2 class labels.

The first convolutional layer filters the 16 x 16 input image block with 64 filters of size 5 x 5 with a stride of 1 pixel and padding 2 producing 64 feature maps of size 16 x 16. Each convolution layer is followed by a ReLu (Rectified Linear Units) layer to introduce non linearity. Max pooling layer follows the first convolutional layer. The second convolutional layer takes as input the pooled output of the first convolutional layer and filters it with 64 filters of size 5 × 5 × 64. The third convolutional layer takes input from the second convolutional layer and filters it with 256 kernels of size 5 × 5 × 64. The output is forwarded to a fully connected layer with 256 neurons followed by

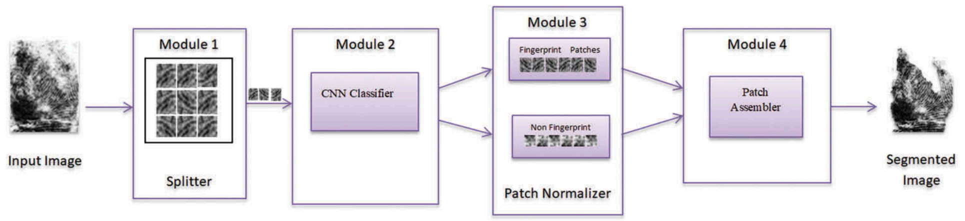

Figure 4. Block diagram of proposed approach.

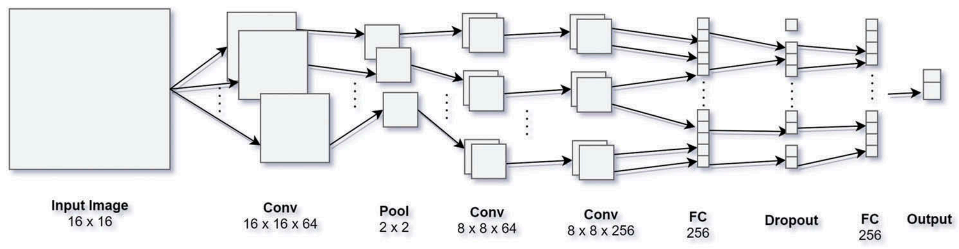

Figure 5. Architecture of proposed CNN model.

dropout layer. Finally, the network has fully connected layer with softmax classifier producing binary classification.

Overfitting

Deep neural networks consist of multiple hidden layers enabling them to learn more complicated features. However, deep neural networks require very large training dataset in order to perform better. With limited training data, many of the complicated features will be the result of sampling noise, which exist in the training set but may not exist in real test data. This leads to the problem of overfitting. Overfitting refers to the problem when a model is trained and it works so well on training data that it negatively impacts the performance of the model on new data. There are many ways to reduce overfitting problem and some of the popular techniques used include data augmentation, normalization, and dropout.

Data augmentation means modifying the current training data in a random way to produce more data. For example images can be slightly scaled, translated, and zoomed to generate more data with same content but with different framing. *Normalization* is the process of scaling down the data to some reasonable limit to standardize the range of features.

Dropout refers to dropping out units (hidden and visible) in a neural network. By dropping a unit out, we mean temporarily removing it from the network, along with all its incoming and outgoing connections. The dropped out neurons neither contribute to the forward pass nor do they contribute in backpropagation (Srivastava et al. 2014). By using dropout, the network is forced to learn more robust features as network architecture changes with each input.

We use normalization in first and second convolutional layer and dropout is used in the last fully-connected layer of the proposed CNN model without which the model shows significant overfitting.

False Patch Removal

Convolutional neural networks have achieved promising results in object detection and recognition tasks but so far almost all the proposed CNN models have been trained and tested on good quality images with less inter-class variation. But not many people have used CNN on complex images like latent fingerprint images. Latent fingerprints images usually are of bad quality due to the presence of overlapping patterns and complex image background, thus making it very tough even for fingerprint experts to distinguish between fingerprint area and non-fingerprint area. The complexity of features increases when poor quality fingerprint images are divided into equal sized small patches. It gets even more difficult to distinguish between fingerprint patch and non-fingerprint patch as these patches have similar features. Figure 6 shows an example of two similar patches from two different classes (fingerprint and non-fingerprint). These patches/blocks look very much similar but are extracted from fingerprint and non-fingerprint areas respectively. Any patch is treated as a false patch if it is misclassified and mislabelled by the classifier and to reduce these false patches, we used majority of neighbors to decide final label of a patch. Since all the patches of a specific class are probably together in same area or neighborhood, the probability of a patch belonging to the majority is higher. For each suspicious block/patch (whose score is less than a defined threshold),

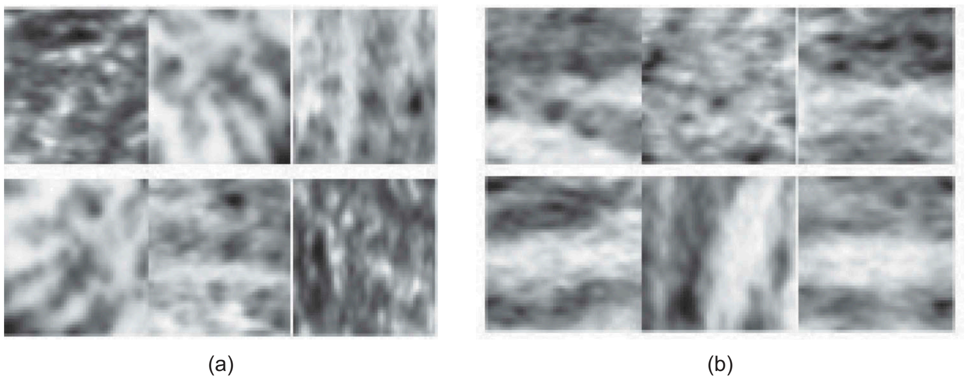

Figure 6. (a) 32 × 32 patches extracted from fingerprint area of a latent fingerprint image from IIIT-D latent database. (b) 32 × 32 patches extracted from non-fingerprint area of a latent fingerprint image from IIIT-D latent database.

its four neighboring patches are observed and if at least 3 neighboring patches are of same class as this patch then it is accepted as true patch otherwise it is treated as false patch and its class label is changed from fingerprint to non-fingerprint or vice versa depending upon the actual class.

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

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

The method "*Majority of neighbors*" performed well on low-quality latent fingerprint images and reduced the false patches by around 20%. The outcome is shown in Figure 7, the figure shows the result of our method prior and after applying "*Majority of neighbors*" technique.

Experimental Results

The proposed algorithm was implemented in MATLAB using MatConvNet-VlFeat (Vedaldi and Lenc 2015) for testing and evaluation purposes.

Latent Database

We used IIIT-D latent database for our research. The database has been published by the Image Analysis and Biometrics Lab Indraprastha Institute of Information Technology, Delhi (Sankaran, Vatsa, and Singh 2012). The database contains 1045 latent fingerprint images from 15 subjects lifted using brush and black powder.

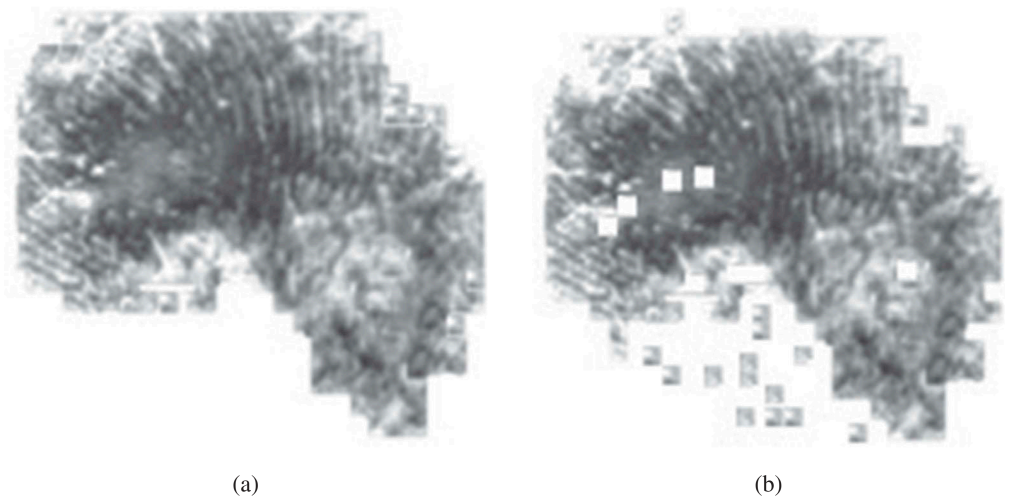

Figure 7. (a) Segmentation result before false patch removal technique. (b) Segmentation result after applying false patch removal technique.

Preparation of Patch Dataset

Latent fingerprint patch dataset was prepared manually by extracting patches of size 16 × 16 and 32 × 32 from latent fingerprint images for training and testing our model. The dataset contains 10,000 patches distributed over two classes namely *fingerprint* and *non-fingerprint*. Few samples of fingerprint and non-fingerprint patches are shown in Figure 8.

Training

We trained our model on 10,000 image patches from IIIT-D latent fingerprint database using stochastic gradient descent (SGD) with learning rate of 0.001, batch size of 50 and weight decay of 1. The weights were initialized using zero-mean Guassian distribution with deviation of 0.01 and scale 1. Figure 9 shows 64 convolutional kernels of size 5 × 5 × 3 learned by first convolutional layer. We trained the network for around

60 epochs and the training took around 23 h on Intel Xenon server with 16 GB RAM and NVIDIA GTX GPU.

Results

We tested our algorithm on IITD Latent Database and the results are shown in tabular form below. Figure 10 shows the result of our proposed segmentation technique on latent fingerprint images. The figure shows 3 input images with their corresponding segmented images obtained after applying our proposed technique. Missed detection rate (MDR) and false detection rate (FDR) of our model was obtained on the patches extracted from the latent fingerprints.

FDR is the rate of non-fingerprint blocks classified as fingerprint blocks and is given as:

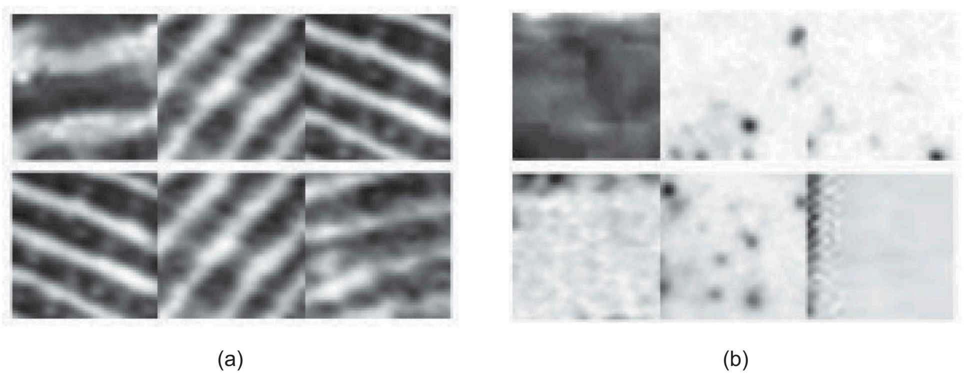

Figure 8. (a) Fingerprint patches. (b) Non-fingerprint patches.

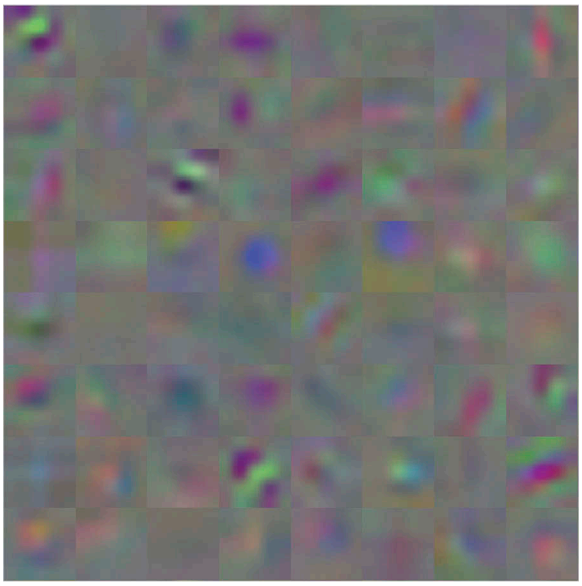

Figure 9. 64 convolutional kernels of size 11 × 11 × 3 learned by first convolutional layer.

$$R = \frac{FP}{FP + NF}$$

Where *FP* is false positives and *NF* is non-fingerprint patches.

MDR is the rate of fingerprint blocks classified as non-fingerprint and is given as:

$$MDR = \frac{FN}{FN + TP}$$

Where *FN* is false negatives and *TP* is total positives.

The results we obtained are summarized in Table 1. The proposed model achieved an accuracy of 83.90% and 94.44% on fingerprint and non-fingerprint patches respectively as shown in Table 1. FDR and MDR achieved was 5.2% and 13.8%, respectively for good, bad, and ugly patches whereas for good quality patches only, the FDR and MDR of the model are 4.7% and 10.5%, respectively.

Afterwards, we compared obtained results with the existing segmentation techniques. A summarized comparison analysis of these techniques is presented in Table 2. The results achieved by our proposed model are superior to most of the existing approaches.

Conclusion

CNN has been continuously surprising us with their achievements in the task of object detection and classification. In this article, we attempted to use CNN for

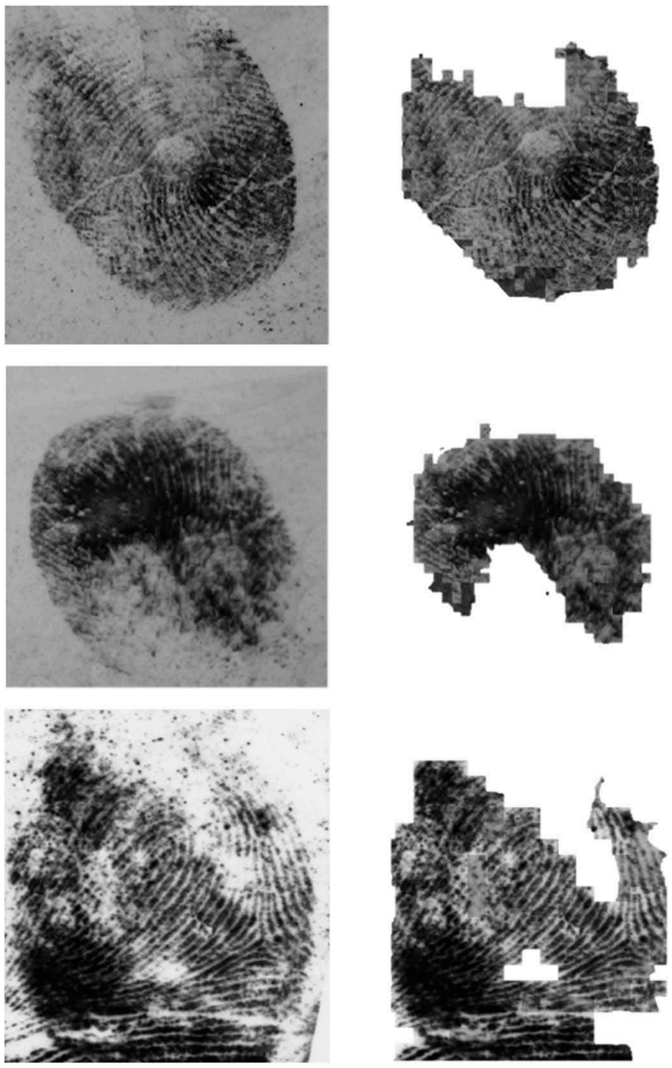

Figure 10. Result of proposed segmentation technique on latent fingerprints from IIIT-D latent database. Left column contains input images and right column their corresponding segmented image.

Table 1. Accuracy of our model on latent fingerprint patches from IIIT-D database.

	Predicted Class		
Actual Class	Fingerprint	Non Fingerprint	Accuracy
Fingerprint	1022	196	83.90 %
Non Fingerprint	63	1065	94.44 %

segmentation of latent fingerprint images. We posed segmentation problem as classification problem by dividing images into patches and then classified these patches into fingerprint and non-fingerprint classes. The proposed technique uses CNN for classification of patches followed by false patch removal technique to

Table 2. Performance comparison of different segmentation techniques.

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

remove false and isolated patches. We tested our proposed segmentation technique on latent fingerprint images from the IIIT-D latent fingerprint database and obtained impressive results. Our future work involves improving classification of latent fingerprints and using deep learning techniques for race-based classification of fingerprints.

References

Arshad, I., G. Raja, and A. Khan. 2014. Latent fingerprints segmentation: Feasibility of using clustering-based automated approach. *Arabian Journal for Science & Engineering (Springer Science & Business Media BV)* 39 (11):12.

Cao, K., and A. K. Jain. 2015. Latent orientation field estimation via convolutional neural network. In Biometrics (ICB), 2015 International Conference on 2015 May 19, 349–56. IEEE.

Choi, H., M. Boaventura, I. A. G. Boaventura, and A. K. Jain. 2012. Automatic segmentation of latent fingerprints. Biometrics: Theory, Applications and Systems (BTAS), 2012 IEEE Fifth International Conference on, 303–10. IEEE.

Ezeobiejesi, J., and B. Bhanu. 2016. Latent fingerprint image segmentation using fractal dimension features and weighted extreme learning machine ensemble. 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, 214–22.

Karimi-Ashtiani, S., and -C.-C. Jay Kuo. 2008. A robust technique for latent fingerprint image segmentation and enhancement. Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on, August, 1492–95. IEEE, 8

Khan, A. I., and M. Arif Wani. 2015. Efficient and rotation invariant fingerprint matching algorithm using adjustment factor. Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on, 1103–10. IEEE.

Khan, A. I., and M. A. Wani. 2014. Strategy to extract reliable minutia points for fingerprint recognition. Advance Computing Conference (IACC), 2014 IEEE International, 1071–75. IEEE.

Krizhevsky, A., I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 1097–105.

Long, J., E. Shelhamer, and T. Darrell. 2015. Fully convolutional networks for semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 3431–40.

Sankaran, A., M. Vatsa, and R. Singh. 2012. Hierarchical fusion for matching simultaneous latent fingerprint. Biometrics: Theory, Applications and Systems (BTAS), 2012 IEEE Fifth International Conference on, 377–82. IEEE.

Sankaran, A., M. Vatsa, and R. Singh. 2014. Latent fingerprint matching: A survey. *IEEE Access* 2 (982):982–1004. doi:10.1109/ACCESS.2014.2349879.

Schmidhuber, J. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61:85–117. doi:10.1016/j.neunet.2014.09.003.

Short, N. J., M. S. Hsiao, A. Lynn Abbott, and E. A. Fox. 2011. Latent fingerprint segmentation using ridge template correlation. Imaging for Crime Detection and Prevention 2011 (ICDP 2011), 4th International Conference on, 1–6. IET.

Srivastava, N., G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15 (1):1929–58.

Szegedy, C., A. Toshev, and D. Erhan. 2013. Deep neural networks for object detection. Advances in Neural Information Processing Systems, 2553–61.

Vedaldi, A., and K. Lenc. 2015. Matconvnet: Convolutional neural networks for matlab. Proceedings of the 23rd ACM international conference on Multimedia, 689–92. ACM.

Wan, J., D. Wang, S. C. H. Hoi, P. Wu, J. Zhu, Y. Zhang, and J. Li. 2014. Deep learning for content-based image retrieval: A comprehensive study. Proceedings of the 22nd ACM international conference on Multimedia, 157–66. ACM.

Zhang, J., R. Lai, and C.-C. Jay Kuo. 2012. Latent fingerprint segmentation with adaptive total variation model. Biometrics (ICB), 2012 5th IAPR International Conference on, 189–95. IEEE.