Fingerprint Image Segmentation using Deep Features and SVM

Reji C. Joy, S. Hemalatha

Abstract: The task of fingerprint segmentation is the most important step in an automated fingerprint identification system. It is essential to separate the fingerprint foreground with ridge and valley structure from the background, which usually contains unwanted data hindering an accurate feature extraction. In the proposed method, fingerprint segmentation is treated as a classification problem by classifying the given input image into foreground class or background class. Here, we have used an unsupervised learning algorithm by using Stacked Sparse Autoencoder (SSAE) to learn the deep features which can very well distinguish the background region from foreground one. Finally, these deep features are given to the SVM classifier. The experimental results prove that the proposed method meets the state-of-the-art results in a wide range of applications.

Index Terms: Autoencoder, fingerprint, morphology, segmentation

I. INTRODUCTION

The advancement in technology and the need to ensure security in various application domains have paved way for the individual recognition and identification system through biometric features – ‘Biometric’ refers to the various automated methods that identify individuals uniquely based on measurable physiological or behavioral features. Among the various biometric features such as iris, face, fingerprint, voice, gait etc., fingerprint is the most common and popular. One reason is that sensors for fingerprint are less costly and the other is that it is the sole mark left by criminals after an offence. An accurate automatic personal identification is critical in application domains such as electronic-commerce, automatic banking and national ID cards. So, an identification system which uses a fingerprint as a biometric is gaining importance.

Each fingerprint is a distinctive arrangement of curved lines called ridges amid two adjacent curved lines or a valley. A captured impression of the finger has two portions: the foreground part and the background part (see Fig. 1). When the tip of any finger is pressed on the scanner surface the image of the foreground is obtained whereas the background is the unwanted noise usually found at the image border. Fingerprint matching is done by extracting feature points from the acquired image. The most commonly used feature points are minutiae and singular points.

The feature extraction algorithm extracts a lot of features from the foreground as well as false features from the background.

Scores of studies have been mentioned in the literature associated with fingerprint image segmentation. Bezen [1] has proposed a method which uses local pixel features like local mean, local variance and local coherence. Later these feature sets are linearly combined to obtain the desired result. The drawback of this method is its low speed. In [2], Gabor features are extracted from the given input fingerprint image. This technique could very well extract Gabor feature from the foreground area. The drawback of this method is that it is not economically feasible. A combination of local mean and local variance of gradient magnitude are extracted from the given input fingerprint image. This technique could very well extract Gabor feature from the background area. The drawback of this method is that it is not economically feasible. A combination of local mean and local variance of gradient magnitude is used for fingerprint image segmentation [3]. In [4], a novel 2D Feature Extraction technique for fingerprints using minutiae points and their intersections has been proposed. Finally the extracted features are given to an SVM classifier. In [5], a neural network layer called MaxPoolingFragment is constructed, and later it is followed by a back-propagation procedure. It is a fast algorithm which segments the images by training MaxPooling Convolutional Networks. In [6], a modified SVM-based method is proposed based on the properties of support vectors, which simultaneously eliminate redundant training vectors. This method works well on real images by reducing the number of input training vectors, while preserving the support vectors. The advantage of this method is its low computational cost.

In this paper, a novel method for fingerprint segmentation by using self-taught deep features is presented. Self-taught learning technique is useful in many...
image processing areas. Self-taught learning can learn to extract high-level features from the fingerprint image. In this work, we have used stacked sparse autoencoder to let the algorithm learn the features automatically from the fingerprint image. These deep features describe great discriminating power to distinguish the foreground and background regions from the fingerprint image. To segment the regions, these deep features are fed into the support vector machine classifier. Post-processing technique using morphological operators is then applied to determine the optimal fingerprint ridge like regions.

The whole work is divided into four sections. Section 2 describes a feature extraction method used for segmentation, Section 3 the system flow and architecture of the proposed fingerprint image segmentation method, Section 4 the test results along with the performance analysis and Section 5 gives the conclusion.

II. FEATURE EXTRACTION

A. Fingerprint as Texture

The oriented flow of ridges and valleys in a fingerprint image introduces intensity changes to form a typical textural pattern [17]. This unique characteristic of a fingerprint image often results in a spatial relationship with local image intensities. Deep neural network architecture can identify these spatial relationships and can extract feature patterns that can discriminate fingerprint foreground from its background.

B. Basic Autoencoder

A basic autoencoder is designed to encode the given input x in the hidden representation h using an encoding function and outputs the reconstructed input using a decoder function. For this, a network structure consisting of an input layer, hidden layer and an output layer is made use of. Therefore, an autoencoder is an unsupervised learning algorithm because it does not use labels for training.

Given and input \( x \in \mathbb{R}^1 \) the encoder function takes the form:

\[
    h = S(Wx + b) \tag{1}
\]

where \( S(\cdot) \) is a non-linear activation function, such as a sigmoid function, \( W \) is a \( 1 \times q \) weight matrix and \( b \in \mathbb{R}^1 \) represents the encoding bias vector.

A decoder function takes the above hidden representation \( h \) and outputs the reconstructed input \( \hat{x} \). It has the form:

\[
    \hat{x} = S(W'h + b') \tag{2}
\]

where \( W' \) and \( b' \) are the weight and bias parameters to the decoder function respectively.

Reconstruction errors can be found using either the squared error \( L(x, \hat{x}) = ||x - \hat{x}||^2 \) in the case of linear reconstruction or the cross-entropy \( L(x, \hat{x}) = -\sum_{i=1}^{q} x_i \log(\hat{x}_i) + (1 - x_i) \log(1 - \hat{x}_i) \). Here, the objective of the autoencoder training is to find the parameter \( \theta = \{W, b, W', b'\} \) that can minimize the reconstruction error on the given training set. This objective function is given by

\[
    J_{AE}(\theta) = \frac{1}{2} \sum_{x \in X} (L(x, \hat{x})) \tag{3}
\]

where \( X \) is the training set and \( N \) the number of samples. A back-propagation optimization algorithm can be used to train the autoencoder.

Since we are using the autoencoder as a tool for feature extraction, this trivial identity mapping does not provide the discriminating patterns to classify the fingerprint foreground and background effectively. However, by placing constraints on the network, such as by limiting the number of hidden units, the autoencoder starts to learn a compressed representation and leads to discover interesting patterns in the input. The basic autoencoder can also introduce over fitting of data. This effect can be reduced by adding a weight-decay regularization term into the objective function to make smaller weights. Therefore the objective function becomes

\[
    J_{AE}(\theta) = \frac{1}{2} \sum_{x \in X} (L(x, \hat{x})) + \frac{\lambda}{2} \sum_{ij} W_{ij}^2 + W_{ij}'^2 \tag{4}
\]

where \( \lambda \) is the regularization term. If the input were completely random then the above compression task would be very difficult. However, since the textures are highly correlated in a fingerprint image, the above model will be able to learn some of these correlations as the feature for fingerprint segmentation.

C. Sparse Autoencoder

Sparsity constraint can also be imposed on the basic autoencoder which can be forced to discover useful features from the input data [7]. These types of autoencoder where the sparsity constraints are added in the hidden nodes, is known as sparse autoencoder. In a neural network model, a neuron is considered to be active if the output value reaches 1 and is inactive if the value reaches 0 [8], [9].

Let the average activation of the hidden layer \( j \) be

\[
    \hat{p}_j = \frac{1}{m} \sum_{i=1}^{m} h_{ij} \tag{5}
\]

We would like to enforce the constraint below approximately

\[
    \hat{p}_j = p_j \tag{6}
\]

where \( p \) is a sparsity parameter, which is a value close to 0. To satisfy this constraint, we will be adding a penalty term to the objective function based on Kullback-Leibler (KL) divergence that can be calculated by

\[
    \sum_{j=1}^{q} KL(p \| \hat{p}_j) = \sum_{j=1}^{q} p \log \frac{p}{\hat{p}_j} + (1 - p) \log \frac{1 - p}{1 - \hat{p}_j} \tag{7}
\]

The final objective function will be of the form

\[
    J_{AE}(\theta) = \frac{1}{2} \sum_{x \in X} (L(x, \hat{x})) + \frac{1}{2} \sum_{ij} W_{ij}^2 + W_{ij}'^2 + \beta \sum_{j=1}^{q} KL(p \| \hat{p}_j) \tag{8}
\]
where $\beta$ controls the weight of the sparsity penalty term.

**D. Stacked Sparse Autoencoder**

It is a variant of neural network with multiple layers of sparse autoencoder (SAE). Here, the output of one layer forms the input of the adjacent layer [7]. By stacking all the layers, a stacked autoencoder behaves like a deep network to identify some complex features in the input data. In this work, we used two hidden sparse autoencoder networks connected with an input layer and an output layer. Finally, all the different layers are stacked to form a deep network. The architecture of SSAE is shown in Fig. 2. As we are interested only in the feature extraction, the decoder parts of the autoencoder are not represented.

![Fig. 2 Structure of Stacked Sparse Autoencoder](image_url)

**III. THE PROPOSED METHOD**

The idea of fingerprint segmentation is to classify the foreground and background regions in a fingerprint image. Therefore, in this work, we treat the fingerprint segmentation task as a classification problem. The proposed method is designed in two phases. In the first phase a deep autoencoder, a self-taught machine learning mechanism, is used for feature extraction. These extracted features contain more subject-specific representations when compared to traditional feature extraction methods. In the second phase, these high-level features feed into a supervised learning algorithm for the possible classification of fingerprint regions. Many general supervised learning techniques [10] as well as methods specific to fingerprint segmentation [1] are proposed in the literature. However, in our work we have used Support Vector Machine (SVM) classifier, since the fingerprint segmentation problem is a strict two-class problem. In the coming section we briefly introduce SVM.

**A. Support Vector Machine**

The core idea of SVM is to map the input data with labels $\{X_i, y_i\}$ where $X_i \in \mathbb{R}^n$ and $y_i \in \{-1, 1\}$, into a higher dimensional feature space by determining a linear separating hyperplane with a large margin to classify the two classes in the training phase [14].

In the testing phase, when new test data are given, SVM uses the following decision function for deciding which class the test data belongs to.

$$f(X) = sgn \sum_{i=1}^{N} a_i y_i \cdot K(X, X_i) + b$$  \hspace{1cm} (9)

where $N$ is the number of data points in the test input and $a_i$ are the coefficients of the trained data set that describes the separating hyperplane. Here, $a_i > 0$ are called support vectors. When it is hard to find a separating hyperplane to separate the input test data, SVM can use a non-linear transformation $\Phi(\cdot)$ to map the test data to a more complex space. $sgn(\psi)$ is the sign function:

$$sgn(\psi) = \begin{cases} 
1, & \text{if } \psi \geq 0, \\
-1, & \text{otherwise}
\end{cases} \quad (10)$$

**B. Stacked Sparse Autoencoder with SVM**

In this paper, we introduce SSAE to generate features directly from the image patches as the feature vector for classification. The size of an image patch is decided to include as many fingerprint textural content as to minimize the chance of both foreground and background occurring in the same patch. In our SSAE network, we used an input layer to feed image patches and two hidden sparse autoencoder to get the hidden representation and these layers are stacked to form a deep network. Therefore, our deep network, pre-trained initially with fingerprint image patches as input, outputs a high-level feature representation as deep feature. The deep feature with the labeled data is given as the input to SVM for the final classification.

In the first phase of training, we used greedy layer-wise pre-training strategy in deep-learning, to generate the deep feature. The architecture of this learning process is illustrated in Fig. 2. The first SAE gets the image patches in the training set $X$ from the input layer to learn primary feature representation $h_1^{(1)}(X)$ by adjusting its weight $W^{(1)}$ value and produces the first layer feature (Features 1). These primary features are then given to the second SAE which produces the higher level features $h_2(X)$ in the output layer. Since this strategy is an unsupervised learning technique, the higher level feature, also called the deep feature [16], are not associated with any labels. In
the second phase, the deep features combined with the labeled data $y_i$, i.e., $\{x_i, y_i\}$ are given as the input feature to a supervised learning algorithm, such as SVM, for the final classification.

C. Morphological Operations

Segmenting the fingerprint image accurately into foreground and background largely depends on the quality of the input image (see Fig. 3(a)).

If the classifier is hard to determine the actual class accurately, segmentation results in producing some spurious regions of one class inside another class. That is, a background class region may appear inside the foreground class regions and vice versa (see Fig. 3(b)). However, these spurious regions can be removed, by treating the smaller regions as part of the larger regions that they are part of, to form a meaningful cluster. In this work, morphological operations [11] are used as the post-processing technique to obtain the best cluster regions. An impact of post-processing is very well clear from Table 2, where 2.56% is misclassified before applying the morphological operations. The morphological operations make use of dilation and erosion operations for removing these spurious regions [15]. We have used a morphological method proposed in our previous work [12] as the post-processing method for determining the optimum cluster regions. The final result of segmentation after applying the morphological operations is shown in Fig. 3(c).

IV. EXPERIMENTAL RESULTS

The proposed fingerprint segmentation algorithm is tested in the standard FVC 2002 Db2_a dataset [10] for its performance evaluation. This dataset consists of 800 images of 8 different fingerprint impressions of 100 individuals, each fingerprint image being of $500 \times 500$ resolution. Each fingerprint image is divided into $28 \times 28$ image patches manually, and 10 image patches are randomly selected from the foreground and background area for training. In the same way 5 image patches are selected for testing. Hence a total of 8000 image patches have been selected for training and 4000 image patches for testing.

The actual performance of SSAE deep network is determined by the different values given to various parameters like the number of hidden layers, sparsity parameter $p$, the weight of the sparsity penalty term $\beta$ and weight-decay regularization term $\lambda$. As there are no thumb rules for identifying the optimal value of these parameters, we have used random methods for selecting the parameters [13]. Our SSAE deep network is designed in such a way that the input layer accepts data of the size $28 \times 28$ image patches and outputs the feature vector of the size 50 for each set of input data. This feature vector is given as the input to SVM. We have used 5-fold cross-validation on the training set in the training phase.

The results of the proposed segmentation algorithm are compared to calculate the misclassification to quantify the algorithm performance. The misclassification can be quantified as:

$$p(f_0|f_1) = \frac{b_{err}}{b_n}$$

$$p(f_1|f_0) = \frac{f_{err}}{f_n}$$

$$p_{err} = \frac{p(f_1|f_0) + p(f_0|f_1)}{2}$$

where $f_{err}$ is number of blocks that are true foregrounds, but erroneously identified as backgrounds by the algorithm, $f_n$ is the number of true foreground blocks that are manually identified from the image and $p(f_1|f_0)$ gives the probability of foreground classification error. $b_{err}$ is the number of blocks that are true backgrounds, but erroneously identified as foreground by the algorithm, $b_n$ is the number of true background blocks that are manually identified from the image and $p(f_0|f_1)$ represents the probability of background classification error.
To quantify the classification accuracy of the proposed algorithm, a sample of 10 random fingerprint images were selected from Db2-a dataset using the equation Eq.11. Further, the morphological operations are consolidated in Table 2. It summarizes the total error probabilities before and after morphological operations on the same 10 fingerprint images. The results show that the overall misclassification rate is very low. (see Fig. 4.)

| Fingerprint | \( f_0 \) | \( b_0 \) | \( f_{err} \) | \( b_{err} \) | \( p(f_1 | \tilde{f}_0) \) | \( p(f_0 | \tilde{f}_1) \) | \( p_{err} \) |
|-------------|----------|----------|-------------|-------------|----------------|----------------|-----------|
| 4_8.tif      | 618      | 221      | 12          | 7           | 0.0317         | 0.0194         | 0.0256    |
| 10_1.tif     | 533      | 231      | 10          | 8           | 0.0346         | 0.0188         | 0.0267    |
| 14_3.tif     | 544      | 156      | 7           | 3           | 0.0192         | 0.0129         | 0.0161    |
| 26_1.tif     | 413      | 178      | 8           | 4           | 0.0225         | 0.0194         | 0.0210    |
| 32_5.tif     | 378      | 287      | 4           | 10          | 0.0348         | 0.0106         | 0.0227    |
| 49_2.tif     | 463      | 197      | 11          | 7           | 0.0355         | 0.0238         | 0.0297    |
| 51_1.tif     | 398      | 312      | 6           | 14          | 0.0449         | 0.0151         | 0.0300    |
| 61_6.tif     | 523      | 275      | 12          | 6           | 0.0218         | 0.0229         | 0.0224    |
| 72_1.tif     | 584      | 195      | 14          | 5           | 0.0256         | 0.0240         | 0.0248    |
| 77_2.tif     | 627      | 265      | 17          | 9           | 0.0340         | 0.0271         | 0.0306    |

| Before Morphology | \( f_0 \) | \( b_0 \) | \( f_{err} \) | \( b_{err} \) | \( p(f_1 \mid \tilde{f}_0) \) | \( p(f_0 \mid \tilde{f}_1) \) | \( p_{err} \) |
|-------------------|----------|----------|-------------|-------------|----------------|----------------|-----------|
| 5081              | 2317     | 101      | 73          | 3           | 0.0315         | 0.0198         | 0.0256    |
| After Morphology  | 5081     | 2317     | 48          | 33          | 0.0142         | 0.0094         | 0.0118    |

One among the many important features of the fingerprint image for any AFIS is the singular points, also known as core and delta points. Singular points are the unique landmark in the fingerprint image that is useful for fingerprint classification and matching. However, singular points especially the delta points are present very close to the image boundary. It is very important for any segmentation algorithm not to miss out on these points in the segmentation process. Therefore, a test is performed in the FVC2002 Db2-a dataset [10] to see if the foreground region contains these feature point after segmentation using our proposed algorithm.

V. CONCLUSION

A new approach is introduced for fingerprint image segmentation using deep features and SVM. Here, we have used stacked sparse auto encoders to learn deep features which are later given to an SVM classifier. The whole performance of any AFIS is greatly influenced by the results of this pre-processing step which gathers reliable features. In the first phase, a greedy layer-wise training approach is used to learn deep features. Then in the second phase, the deep features are processed by the SVM-supervised classifier. Experimental results show that the input fingerprint image achieves the state-of-the-art results in all domains of applications.

REFERENCES
1. A. Bazen and S. Gerez, "Segmentation of fingerprint images," in ProRISC 2001 Workshop on Circuits, Systems and Signal Processing, Veldhoven, TheNetherlands, 2001.
2. F. Alonso-Fernandez, J. Fierrez-Aguilar and J. Ortega-Garcia, "An enhanced Gabor filter-based segmentation algorithm for fingerprint recognition systems," in Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA2005), 2005.
3. Saparudin, "Segmentation of Fingerprint Image Based on Gradient Magnitude and Coherence," IECE, 2015.Nibras, "A Novel 2D feature extraction Method for Fingerprints Using Minutiae Points and Their Intersections," IECE, 2017.
4. M. Jonathan, "A fast learning algorithm for image segmentation with max-pooling convolutional networks," arXiv:1302.1690v1[cs.CV], 2013.
5. Y. Zhiwen, "A modified support vector machine andits application to image segmentation," Image and Vision computing, 2011.
6. N. Andrew, "Cs294a lecture notes: Sparse autoencoder," 2010. [Online].
7. N. Andrew, "Sparse Autoencoder," 2011. [Online]. Available: https://web.stanford.edu/class/cs294a/sparseAutoencoder_201Inew.pdf . [Accessed 22 10 2017].
8. N. Andrew, "Sparse Autoencoder," 2017. [Online]. Available:
Fingerprint Image Segmentation using Deep Features and SVM

http://web.stanford.edu/class/archive/cs/cs294a/cs294a.1104/sparseAutoencoder.pdf. [Accessed 22 10 2017].

9. D. Maltoni, D. Maio, A. Jain and S. Prabhakar, Handbook of fingerprint recognition (Second Edition), New York: Springer, 2009.
10. R. Gonzalez and P. Wintz, Digital Image Processing.2nd Edition, Addison-Wesley, 1987, pp. 153-160.
11. C. Reji and S. Hemalatha, “A Gradient Based Approach for Fingerprint Image Segmentation using Morphological Operators,” International Journal of Engineering & Technology, 2018.
12. C. Gravelines, “Deep learning via stacked sparse autoencoders for automated voxel-wise brain parcellation based on functional connectivity,” Electron. Thesis Diss. Repos., pp. 1-75, 2014.
13. V. Vapnik, Statistical Learning Theory, New York: Wiley, 1998.
14. P. Soille, Morphological Image Analysis: Principles and Applications, Springer-Verlag, 1999.
15. Y. Bengio, "Learning deep architectures for ai," Found. Trends. Mach. Learn., vol. Vol.2, pp. 1-127, 2009.
16. C. J. Reji and M. Azath, "Fingerprint Image Segmentation using Textural Features," in Proceedings of International Conference on Computer Vision and Image Processing, Springer Nature, 2017.

AUTHORS PROFILE

Reji C Joy is a research scholar at Department of Computer Science, Karpagam Academy of Higher Education, Coimbatore. She obtained her Master of Computer Applications from Amrita Institute of Science and Technology, Coimbatore, Tamil Nadu (2004). Her researches are in fields of image processing. Her work basically includes fingerprint image segmentation.

S. Hemalatha completed MCA, PhD in Computer Science and currently working as Assistant Professor in Department of Computer Science in Karpagam Academy of Higher Education, Coimbatore,. She has ten years of experience in teaching and presented papers in National and International Conferences, Journals and Magazines. Her area of research is Data Mining, Image Processing and Networks.