IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Recreating Fingerprint Images by Convolutional Neural Network Autoencoder Architecture

Sergio Saponara[1], Abdussalam Elhanashi[1], and Qinghe Zheng[2]
[1]Dip. Ingegneria dell'Informazione, University of Pisa, Italy
[2]School of Information Science and Engineering, Shandong University, China

Corresponding author: Sergio Saponara (sergio.saponara@unipi.it).

This work was supported by MIUR-Dipartimento di Eccellenza Crosslab Project at university of Pisa.

ABSTRACT Fingerprint recognition systems have been applied widely to adopt accurate and reliable biometric identification between individuals. Deep learning, especially Convolutional Neural Network (CNN) has made a tremendous success in the field of computer vision for pattern recognition. Several approaches have been applied to reconstruct fingerprint images. However, these algorithms encountered problems with various overlapping patterns and poor quality on the images. In this work, a convolutional neural network autoencoder has been used to reconstruct fingerprint images. An autoencoder is a technique, which is able to replicate data in the images. The advantage of convolutional neural networks makes it suitable for feature extraction. Four datasets of fingerprint images have been used to prove the robustness of the proposed architecture. The dataset of fingerprint images has been collected from various real resources. These datasets include a fingerprint verification competition (FVC) database, which has been distorted. The proposed approach has been assessed by calculating the mean square error between the reconstructed and the original features. The trained architecture was tested and compared to the other state-of-the-art methods. The achieved experimental results show that the proposed solution is suitable for recreating a complex context of fingerprinting images.

INDEX TERMS Fingerprint images, Convolution Neural Networks, Autoencoder, Feature extraction, System identification.

I. INTRODUCTION

Nowadays, biometric technology has been widely used in various authentication occasions in industrial and everyday life applications, including mobile payment [1], security verification [2], smart home [3] and so on. The system that can recognize humans is designed using the physical characteristics (e.g., fingerprint [4] and retina [5]) or behavioral characteristics like voice [6] and gait [7]. Among them, fingerprints are the most widely used biometric, with the property of uniqueness, invariability, and high security. Moreover, the acquisition of fingerprints is convenient, which makes fingerprint identification technology widely used in embedded applications.

With the continuous upgrading of chip manufacturing and other processes, the collection area of the fingerprint collector has become smaller, so that the area of the collected fingerprint image is correspondingly reduced, and the fingerprint image is easily destroyed. Due to the influence of fingerprints themselves (dry, wet, dirty, cocoon, scars, etc.) and various collection equipment (dirty collection head, low-resolution, signal transmission noise, etc.), there are a lot of low-quality fingerprint images in actual fingerprint recognition. In general, we are faced with challenges in terms of poor image quality, unclear texture, nonlinear distortion, matching methods, and public potential fingerprint databases [8]. High computational efficiency [9][10] is also a key requirement for the application of fingerprint-related techniques for mobile devices. Furthermore, complicated overlapping patterns will also lead to low quality fingerprint images, which seriously affects the accuracy of the automatic fingerprint identification system [11].

Specifically, low quality fingerprint images can lead to the following problems:

- A large number of pseudo feature points cause serious interference to the recognition system.
- The loss of true feature points leads to mis-training of the recognition model.
- Due to the complex changes in the perspective and posture of the object between different images, it may lead to inaccurate estimation of position and movement direction of the feature points in low-quality images, resulting in a large deviation from the real result.

Since many fingerprint recognition algorithms [12, 13, 14] rely on minutiae features, the minutiae of small-area fingerprint images cannot support the algorithm for differential matching. Therefore, the restoration of low-quality fingerprint images is an urgent problem to be solved during the process of fingerprint recognition and matching.

In recent years, many researchers have conducted in-depth research on fingerprint enhancement or recovery technology. Fingerprint image recovery can be divided into two categories: spatial domain enhancement and frequency domain enhancement. Spatial domain enhancement methods include the directional filtering [15], Gabor filtering [16], and partial differential equation filtering [17]. The frequency domain enhancement methods consist mainly of Fourier domain enhancement [18], short-time Fourier transform enhancement [19], wavelet transform enhancement [20], discrete cosine transform enhancement [21] and so on.

Fingerprint images present unique texture features, which are essentially two-dimensional non-stationary signals. The commonly used median filters or low-pass filters in the image processing can reduce the noise and distortion in the image, but their effect is not ideal because they uniformly process all the pixels indiscriminately. The key to Gabor filtering is how to get the ridge period accurately and quickly, otherwise the filtered fingerprint image will appear empty [22]. Therefore, a good fingerprint reconstruction algorithm can adaptively use the local frequency information and the ridge direction information to enhance the ridge and valley structure, to better distinguish the ridge and valley.

At present, deep learning [23] has been widely used in image processing and other fields. Thanks to the efficient fitting capability of massive parameters, it can usually well capture the data distribution structure and the characteristics of the data itself. However, it is difficult to apply typical deep learning models directly to image reconstruction, especially for fingerprint images with high requirements for detail capture. This means that traditional deep learning models are difficult to generalize well under various conditions. On the other hand, the reasoning speed of a typical deep learning model may be difficult to meet the real-time requirements of practical applications. In this situation, it is critical to develop a lightweight neural network model that can capture fingerprint features.

The main contribution and motivation of this work are:

- Analysis using two different autoencoders (sparse, and convolutional neural network models) on fingerprint classification to examine their robustness for extracting complex context features, which can improve fingerprint recognition.
- Large number of experiments performed, using large scale of fingerprint dataset, aiming to obtain further insights into the performance of CNN autoencoder on different datasets with varied fingerprint features.
- Utilization of light-weight neural network architectures to perform competitive classification accuracy with few parameters for the fingerprint recognition on the images, while having less computation costs than the existing pre-trained neural network architectures.

In this paper, a convolutional neural network (CNN) Autoencoder is used to reconstruct fingerprint images. We will explore the effectiveness on light-weight CNN architecture on replication the complex fingerprint features from the images in comparison to the other deep learning models, and the state of art methods. In this research, we will explain how the CNN autoencoder will be built, and the objectiveness of learning fingerprint features representation of input data from the images into the output data.

Hereafter, the paper is organized as follows: Section I and Section II deal with an introduction and related work. Section III presents deep learning for image reconstruction. Section IV explores the methodology and discusses the global architecture. Section V shows the results and discussion. Conclusions and other experimental targets are drawn in Section VI.

II. RELATED WORK

In this section, we present the related work of fingerprint image recovery and identification from the following two respects: the traditional scheme based on filtering, and the deep learning scheme based on feature description.

A. FILTERING BASED TRADITIONAL SCHEMES

Chakraborty and Rao [24] proposed the fingerprint image enhancement method based on adaptive filtering in frequency domain. The histogram equalization process is performed on the fingerprint image after Gabor filtering, and the enhancement effect of the original fingerprint image can be obtained. In view of the high computational complexity of Gabor filter, Chen et al. [16] proposed to decompose the unrotated two-dimensional Gabor filter into one-dimensional band-pass Gabor filter and 1D low-pass Gabor filter. Chen et al. [17] improved the second-order oriented partial differential equations (PDE) model for fingerprint image restoration, which can connect broken fingerprint ridges, fill in the holes of the fingerprint image, smooth irregular ridges, and eliminate some annoying small flaws. Mei et al. [18] proposed to use curve area transformation in the Fourier domain, find the curve area and map it to a two-dimensional array, and design a filter to restore the fingerprint image based on the frequency image of the curve areas. Ghafoor et al. [19] proposed a frequency distortion elimination and enhancement method based on short-time Fourier transform analysis (STFT) and local adaptive context filtering and verified the superiority of the algorithm through a large number of experiments. In fact, the spatial and temporal resolution of STFT cannot be considered at the same time, which can be solved well by wavelet transform or wavelet packet transform. Moreover, the performance improvement of hyper-parametric optimization of STFT can also be further discussed, such as Fourier number, sampling frequency, window length, etc. Aiming at the problems of cracks, scars, dry skin, poor contrast between ridges and valleys in low-quality fingerprint images, Bidishaw et al. [20] proposed an effective two-stage block enhancement scheme, learning in the space and frequency domain of the basic image. Liu et al. [21] proposed a method to reconstruct the fingerprint orientation field using weighted discrete cosine transform. Ding et al. [22] used the classification dictionary learning to enhance fingerprint image based on spectral diffusion. Although many schemes based on filtering have been proposed for fingerprint image restoration, they still have the problem that the accuracy and efficiency are difficult to meet at the same time.

Yoon et al. [11] studied the multi-layer statistical model and covariates of fingerprint matching (similarity) score analysis and showed that the quality difference between two fingerprints compared will greatly affect the time stability of the fingerprint identification accuracy. To solve the problem of the lack of minutiae features in the fingerprint area, Deshpande et al. [12] proposed the latent minutiae similarity (LMS) algorithm and the clustering latent minutia pattern (CLMP) algorithm, which achieved the best results on multiple datasets. Wang et al. [13] proposed Fin Privacy, a privacy protection mechanism for fingerprint recognition, which injected Laplace noises into the singular values of the approximate singular matrix, thereby weighing privacy and utility. Cao and Jain [14] fused the comparison scores between the potential fingerprints based on the three templates and the reference fingerprints and retrieves a short candidate list from the reference database. By designing the enhancement and the orientation deconvolution branch, an end-to-end deep learning model named FingerNet [25] is proposed for potential fingerprint enhancement. To extend the fingerprint matching technology, A. Manickam et al. [26] proposed the use of Scale Invariant Feature Transform (SIFT) to enhance and match potential fingerprints. Cao et al. [27] proposed an end-to-end latent fingerprint search system, which consisted of an automatic region of interest (ROI) cropping, latent image preprocessing, feature extraction, feature comparison, and an output candidate list.

Compared with the filtering method, the method of convolutional neural network autoencoder proposed in this paper uses statistic knowledge to realize the optimization and application of parametric model, which can adapt to complex situations without prior information. With the increase of the amount of available data, the method based on deep learning has obvious advantages, which benefits from the self-organization of implicit knowledge in the training process.

B. FEATURE DESCRIPTION BASED DEEP LEARNING SCHEMES

Taking defect fingerprints as the object, Wang et al. [28] proposed an improved fingerprint recognition method based on deep CNN with point features. The experimental results show the superiority of deep learning over kernel principal component analysis (KPCA) and k-nearest neighbor (KNN). Aiming at the problems of fingerprint rotation, scaling, damage, Wang et al. [29] proposed a robust fingerprint recognition method based on CNN, which is not only fast but also has a high ability to resist abnormal degradation. A deep learning based unique affine Fourier moment matching (AFMM) method [30] is proposed to match and fuse the scores obtained from three different fingerprint features to deal with local and global linear distortion. Pandya et al. [31] proposed a new deep learning architecture for fingerprint recognition, which achieved 98.21% classification accuracy with only a loss of 0.9. Li [32] empirically proved that the improved CNN recognition method has fewer iterations during the training process and the training error is also small; when identifying unknown fingerprints, the improved CNN method has a lower false recognition rate and rejection rate.

Deep CNNs can learn discriminative features from original fingerprint images instead of explicit feature extraction, which makes them attractive in fingerprint identification. Zia et al. [33] used the uncertainty of a Bayesian model to reduce the number of false positives of fingerprints to improve identification efficiency. After obtaining the number of quality improvement processes needed for fingerprint images, the deep CNN model combined with batch normalization technology was used [34]. Peralta et al. [35] proposed a method that combined image processing with a CNN classifier for fingerprint identification, avoiding the necessity of explicit feature extraction. Aiming at the problem that the traditional fingerprint recognition algorithm relies too much on the details of fingerprint and the recognition performance is limited in mobile devices, Zeng et al. [36] proposed a local fingerprint recognition method based on deep learning. By improving the structure of CNN, two loss functions are optimized, and the identification performance of fingerprint image is improved.

Although deep learning has shown great advantages in the pattern recognition field, it still faces many challenges. One is that how researchers can know that a model still has a good generalization ability for scenes that have never appeared before [37]. Another difficulty is how to make better use of small-scale training data [38] and multi-model data [39]. The processing method during data transmission may also affect the data [40].

The proposed lightweight convolutional autoencoder structure is different from the traditional autoencoder or convolutional neural network. It can reduce the parameter scale while ensuring the feature extraction ability and data representation function, which is conducive to improve the reasoning speed of the model in the process of practical application.

III. DEEP LEARNING FOR IMAGING RECONSTRUCTION

A. DEEP LEARNING FOR FEATURE EXTRACTION

The diffusion of deep learning technologies has paved the way for extracting the features automatically. Several methods were presented to compress data in the images into a lower dimension effectively without a significant loss of data. The enhancements in deep learning for feature extraction are the foundation of remarkable success in

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3124746, IEEE Access

computer vision. Sun et al. [41] explored face recognition with deep learning. This approach is applied to the CNN in order to reduce the dimension of specific regions of the input images, and to obtain a series of deep learning IDs, which are combined together. Deep learning models have been applied as supervised and unsupervised end-to-end regression, and classification algorithms. However, they can be used for feature extraction, and combined with machine learning to treat input complex data efficiently without the requirement of time consuming, poor feature extraction from the images.

B. ARTIFICIAL NEURAL NETWORK

An artificial neural network (ANN) is the principle of deep learning technologies. ANN are the brain since they are applied by combining several simple units, which are called as neurons. ANN models have been improved during the years. More complex architectures, called CNNs, have been deployed in several applications, thanks to their achievements in computer visions. CNNs exploit a multilayer structure which are different from hidden neurons. An ANN provides approximation function which can be defined by the following equation Eq [1]:

$$y = f'*'(x,0) \qquad (1)$$

Where f is the complex arbitrary continuous function, which; parameterized by a set of coefficients [42]. The establishment of the predictive model requires an estimation for several parameters that approximate the targeted output. This can be achieved by minimizing the parameters of cost function for regression and cross-entropy for image classification. Gradient-descent based algorithm is utilized, based on backpropagation algorithm [43].

C. AUTOENCODERS

Autoencoders are models where the algorithm is trained to replicate its own input in an unsupervised way [44]. Autoencoders apply a symmetric structure which includes three main components (encoder, decoder, latent representation), see Figure 1. An encoder part that compresses the input into a low-dimensional representation that contains the context of data. The second part is a decoder, which is trained to reconstruct the features which were extracted by the encoder. Latent representation is one component of the autoencoder, which extracts the relevant information by compressing the information, which traverses the neural network, forcing the learnt information compression of input data from the encoder part. The latent space reduces the dimension and compresses the complexity of the data through a bottleneck.

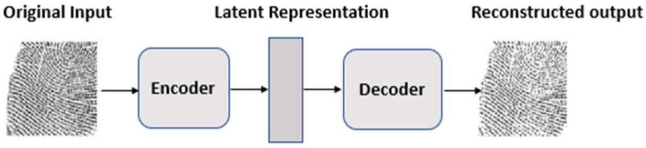


FIGURE 1. Structure of simple autoencoder.

The latent space is determined by X, and visible (data) space by γ assuming they are real valued with dimensionality J and K respectively. The parameters of the autoencoder are optimized jointly in the encoder, and the decoder over the least-squares reconstruction cost. This behavior is formalized in Eq [2]

$$\varphi^*, \psi^* = \underset{\varphi,\psi}{argmin} \sum_{n=1}^{N} \sum_{k=1}^{K} (y_k^{(n)} - f_k(g(y^{(n)}; \varphi); \psi))^2 \qquad (2)$$

In Eq [2] the encoder is represented by ф, the decoder is defined by ψ. $f_k(.)$ is the kth output value of $f(.)$, $argmin$ represents for argument of the minimum, and given N is the amount of data. The model is equivalent to the component's analysis when f, and g are linear. However, the non-linear functions empower for a more robust non-linear mapping. Therefore. Sigmoid function is used as activation function across hidden units. It is a useful property for image data, which makes the learning process more stable for the model.

Recently, autoencoders have become more widely used for learning generative data. The objective of autoencoders is to capture the most important features in the data. There are different kinds of autoencoders which aim to achieve different kinds of applications, which are described as the following:

1. UNDERCOMPLETE AUTOENCODER

This architecture has a three-layer net, i.e., a neural network with hidden layers. The input and the output are the same, and it reconstructs its input to the output by using an Adam optimizer and mean squared error loose function (MSE). The aim of this model is to minimize the loose function by penalizing the g(f(x)) for being varied from its input (x). This autoencoder does not require regularization as it maximizes the probability of data instead of copying the input to the output.

2. SPARSE AUTOENCODER (SAE)

This is a simple autoencoder and is easy to construct. This model has hidden nodes more than the input nodes. The important features are recognized from the given data. Sparsity constraint is used in this model in the hidden layers. This is to prevent the output layers from copying the input data. The hidden layers in this architecture are set at a minimum value to confirm the activation value for the sparsity constraint which is determined as p, and the penalty function is used to prevent ρj variation from the value of p. Kullback-Leibler variation is used as the cost function of the penalty, determined in Eq [3].

$$KL\ (\rho \parallel \rho j) = \rho \ln \frac{\rho}{\rho j} + (1-\rho) \ln \frac{1-\rho}{1-\rho j} \qquad (3)$$

where ρj is not diverging from the parameter p, the Kullback-Leibler value is 0, if not, the Kullback-Leibler value will rise with the divergence.

3. STACKED DENOISING AUTOENCODER (SDAE)

Stacked models are neural networks with multiple layers of sparse autoencoders. In this model, more hidden layers are used, which helps to reduce high dimensional data into a smaller code representing important features from the input data. Each hidden layer in this model is more compact than the last hidden layer. Input corruption is used only in this architecture for initial denoising. It helps to learn the important features from the input data, and once the mapping function f(θ) has learnt, for further layers. the uncorrupted layer is utilized from the previous layers.

4-CONVOLUTIONAL NEURAL NETWORK AUTOENCODER

CNN model is one of deep learning approach, which becomes one of state-of-art for computer vision application due to its significant advantages [45]. Feature learning is one of the advantages of CNN, which it can learn and extract important features. CNN can also learn from a large number of datasets due to its deep architecture. Feature extraction is a crucial issue for pattern recognition, and it is a difficult issue since it depends on the type from the given data [46]. Features are required as representatives for the images. CNN is a deep learning model for feature extraction, which provides self-learning layers. The rationale behind the convolutional neural autoencoder is that the images could be compressed to simple vector, which could be decoded to recreate the original image. The element in the encoder vector does not mean to encode one feature. Since there are millions of parameters in the decoding network, the combination can encode and create a massive number of features. Therefore, convolutional neural autoencoder is implemented to perform unsupervised learning for feature extraction and dimension reduction. As small features are projected to a lower dimension, the distance between the vectors is significantly faster to compute.

A convolutional autoencoder has a structure similar to CNN. It has the same basic components that includes convolutional filters and pooling layers. However, there is difference in the structure of the architecture that both input and output nodes have the same dimension. In regard to this, the reconstructed data can be compared to the input data. The difference between the input data and the reconstructed data can be considered as an algorithm function for the autoencoder thus the learning process is not dependent on the labelled data. A CNN autoencoder is a kind of unsupervised learning architecture. Convolutional neural network (CNN) is a family of deep learning models, which has one or more convolutional layers. It is mainly used for imaging processing and feature extraction from the images. Convolutional autoencoder use convolution operator to encode the input features and replicate them in the output with the minimum reconstruction error. Convolutional autoencoder operation includes m convolutional kernels, and the output layer m feature map. The input feature map is produced from the input layer, n represents the number of input channels. The latent representation for convolutional autoencoder of k-th feature map in the encoder is defined by Eq [4] where σ represents the activation function and * is the two dimensional convolutional. The reconstruction in the decoder is defined by using the following formula see Eq [5], where c represents the bias per the input channel, and H represents the latent feature maps.

$$h^k = \sigma\,(x * W^k + b^k) \tag{4}$$

$$y = \sigma\,(\textstyle\sum_{k\epsilon H} h^k * \widetilde{W^k} + c) \tag{5}$$

CNNs architecture is well-suited for recognition of the objects in the images. To optimize the performance of CNN architecture for a specific application scenario, we need to train, and fine-tune this architecture effectively. Therefore, starting from trained CNN architecture, new data are fed containing unknown classes. Once the network is in place, a new task can be carried out such as fingerprint classification in our case.

IV. METHODOLOGY

A. DATASET DESCRIPTION

Fingerprint image datasets were collected from different resources. In this research, we used four different datasets to assess the effectiveness of autoencoders for replicating the given input data from fingerprint images to their output. These images were detached from the real identity of the individuals and were acquired from different scanners, sensors, and inked devices. The datasets are described as the following:

Dataset I

This dataset has 250 images with size of 200 × 200 pixels. These images were acquired by using the fingerprint device, Digital persona model (4500) reader which were taken from students and faculty staff at YCCE College [47].

Dataset II

This dataset is made up of 250 images with size of 153 × 185 pixels. These images were also collected using the same fingerprint (Digital persona model (4500) reader) device from individuals and members at YCCE College [48].

Dataset III

This dataset was collected from database of Sokoto Coventry Fingerprint Dataset (SOCOFing). It is designed for academic research purposes. This dataset is comprised of 6,000 fingerprint images, belonging to 600 African subjects. It consists of specific features and attributes for both genders, hand, and finger names as well as synthetically modified versions with three different levels of alteration for central rotation, obliteration, and z-cut by utilizing the STRANGE framework. STRANGE toolbox is a novel approach for generation of realistic synthetic alterations on the images of fingerprint. These alterations were performed using simple, medium, and advanced parameter settings in the STRANGE toolbox over 500dbi resolution images. The resolution of these images is 96 × 103 pixels. The images of this dataset were collected by using a scanner, Hamster plus

(HSDU03PTM). This dataset is categorized into three levels of alteration difficulties: easy, medium, and hard [49].

Dataset IV

This dataset consists of 320 images with different sizes of images. These images were collected from fingerprint verification competition database (FVC2004). In FVC2004, we have four databases DB1, DB2, DB3, and DB4 based on the type of the scanner used for acquiring the fingerprint and image size [50]. This dataset has lesser quality for pattern features of fingerprint in comparison to the other datasets. FVC2004 dataset is considered state of the art and the most challenging database due to its perturbations and complex context features in the fingerprint images. These images were mainly designed for training and evaluating deep learning models for pattern recognition purposes. Figure 2 below, shows some samples from four datasets of fingerprint images.

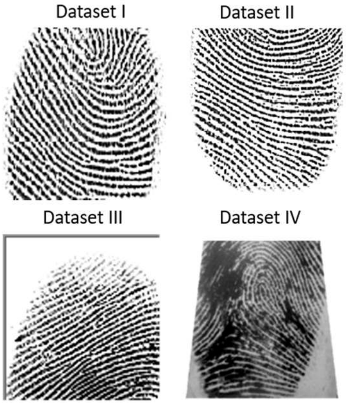


FIGURE 2. Samples for fingerprint images from 4 datasets.

B. DEEP LEARNING MODELS

In the experiments, we used the sparse and convolutional neural autoencoders to obtain the recreated fingerprint images with the best replication to prove the effectiveness of those autoencoders for fingerprint feature recreation.

1. SPARSE AUTOENCODER

We started the experiments with the Sparse autoencoder with the pre-processing activities. The images were split into 80% for training, and 20% for testing for each dataset. The pre-processing has been performed on both training and testing data for each dataset. The image of each dataset was rescaled into 100 × 100 pixels (width and height) for all datasets, and we eliminated the blank space around the fingerprint image itself. This was done in order to achieve an equal number of tiles while cropping the fingerprint images.

We also applied different filters to the image to enhance the sparse autoencoder model's understanding of the structure. We converted the images into grayscale whereas the pattern features of the fingerprints present as a black color, and the background of the images show as white. This is to improve and achieve a binary image for extracting the features. This process minimizes the distortion and the variableness in the fingerprint images, with an outcome of the extraction of beneficial data which also introduces specific artifact features that can affect the stages of pre-processing. As per the requirement of this model, it is important to train and test the architecture with small images, therefore we performed the process of cropping, dividing each image into the tiles, and reassembling these tiles in the form of a reconstructed image. We performed the copping of each single image and created tiles with different sizes (50 × 50, 25 × 25, 20 × 20, and 10 × 10) of pixels to examine the sparse autoencoder architecture with more than one scenario. This pre-processing has been carried by the imaging processing toolbox. Figure 3 shows the pre-processing workflow to prepare fingerprint images to train and test the sparse autoencoder model.

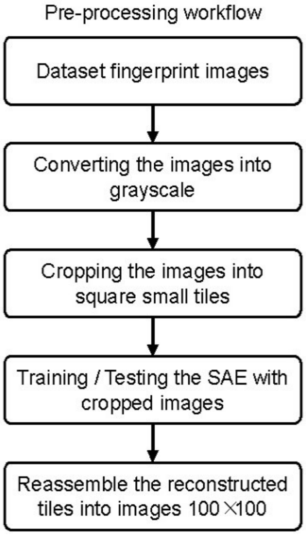

Pre-processing workflow
Dataset fingerprint images
Converting the images into grayscale
Cropping the images into square small tiles
Training / Testing the SAE with cropped images
Reassemble the reconstructed tiles into images 100 ×100

FIGURE 3. The pre-processing workflow to prepare the Fingerprint images to be trained and tested with the Sparse autoencoder.

The Sparse autoencoder has been designed which includes an input layer (encoder), an output layer (decoder) and the latent representation (hidden units), see Figure 4. We set the number of hidden units in the laten representation with 50 neurons. We selected the transfer function for encoder and decoder for this architecture. For encoder part, we chose the linear satlin, and for decoder part, we selected the linear function purlin. Table 1 shows the best hyperparameter selected and used in the training stage. L2 regularization is utilized for training the architecture to overcome overfitting problems. The sparse autoencoder has been trained with four datasets with various sizes of cropped images into tiles with various pixels sizes (50 × 50, 25 × 25, 20 × 20, and 10 × 10). This is to examine and analyze the performance of the sparse autoencoder by comparing the loose function among different sizes of cropped tiles. Our aim was to have the latent representation of the input learnt features of fingerprints from the images by the SAE model and obtain the most minimum mean square error value (MSE), which can be determined as the average of the square for the variation between the predicted and the original values. MSE is an essential algorithm, corresponding with the produced value of mean square error loss. We carried out the training process for the sparse autoencoder and as we can see in Figure 5, the learning curve of the sparse autoencoder enhanced where the size of the cropped tiles is reduced, we achieved the best learning curve for the architecture with a cropped tile size of 10 × 10 pixels.

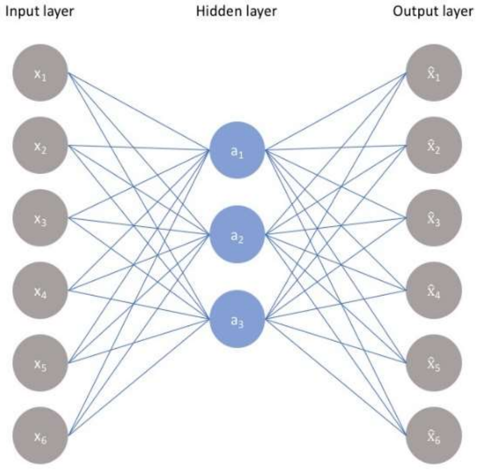


FIGURE 4. Structure of the Sparse Autoencoder (SAE).

TABLE 1. Training Hyper-Parameters for Sparse autoencoder.

Parameter	Method
Number of Epochs	1000
L2 Regularization	0.007
Number of neurons	50
Decoder Transfer Function	Purline
Encoder Transfer Function	Satlin
Sparsity regularization	0.05

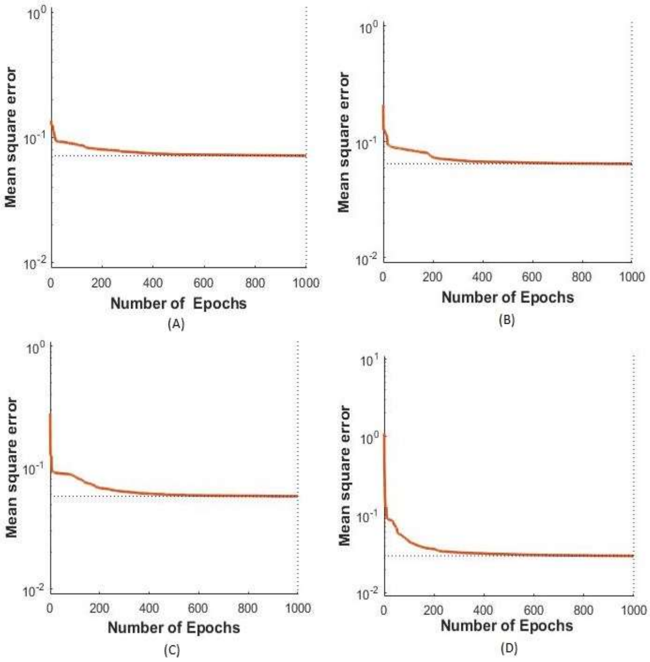


FIGURE 5. Training performance for the sparse autoencoder (SAE) with different sizes for cropped tiles of fingerprint images: A) Tile size (50 × 50), B) Tile size (25 × 25), C) Tile size (20 × 20) and D) Tile size (10 × 10).

Furthermore, we performed the measurement for the values of the mean square error, and we assessed the best value when the sparse autoencoder has been trained with cropped tiles size of 10 × 10 pixels, see Table 2.

TABLE 2. The measured values for mean square error (MSE) for sparse autoencoder with different sizes of cropped tiles during training process.

Image size	Size of each cropped image (Tile)	mseError
100 x 100	50 x 50	0.069
100 x 100	25 x 25	0.061
100 x 100	20 x 20	0.053
100 x 100	10 x 10	0.028

2. CONVOLUTIONAL NEURAL NETWORK AUTOENCODER (THE PROPOSED ARCHITECTURE)

In this section, we will discuss the experiments for recreating the fingerprint images with the CNN autoencoder. As part of preprocessing, the images of four datasets are with pixel values ranging from 0 – 255. These images are resized into 224 × 224 pixels. These 224 × 224 images are converted into matrix 224 ×224 ×1. This is to account the requirement to feed the input data into the convolutional neural network input layer. The images have been randomly split into 70% for training, 20% for validation, and 10% for testing the model for each dataset. It is important to partition the data to generalize the model and reduce the chances of overfitting. The proposed approach includes a set of convolutional and max pooling layers, see Figure 6.

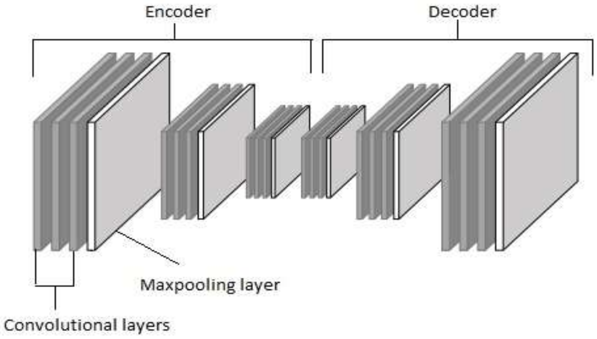


FIGURE 6. Structure of CNN Autoencoder.

We constructed the CNN architecture with 11 layers. This light-weight architecture is established with a low number of CNN layers to account the requirement of IoT and low-cost embedded devices in terms of power consumption and memory usage. The convolutional layers were used to map the features from the input images. The filter size has been set to [3 × 3]. This size is commonly used for the CNNs models. These filters determine the height and the width of the regions in which the neural network connect to the input. Max pooling layers were utilized in this model to down-sample the images into small regions. The filter size for stride has been set in these layers with [2 × 2]. The CNN autoencoder will be split into two parts, which are encoder, and decoder. The first part (encoder) will include the first layer with 32 filters, second layer with 64 filters, and the final layer with 128 filters. The second part (decoder) will include the first layer with 128 filters, second layer with 64 filters, and the final layer with 32 filters. Hyper-parameters have

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3124746, IEEE Access

been set to optimize the training of the CNN autoencoder, see Table 3. We trained the CNN autoencoder with a number of epochs which were set with 1000 for each dataset. These epochs will determine the duration for the training time that the algorithm will work throughout the training dataset. L2 regularization has been fine-tuned with 0.005. We set the batch size with 128 for training the model. Figure 7 shows the training and validation loose curves for the CNN autoencoder throughout all four datasets.

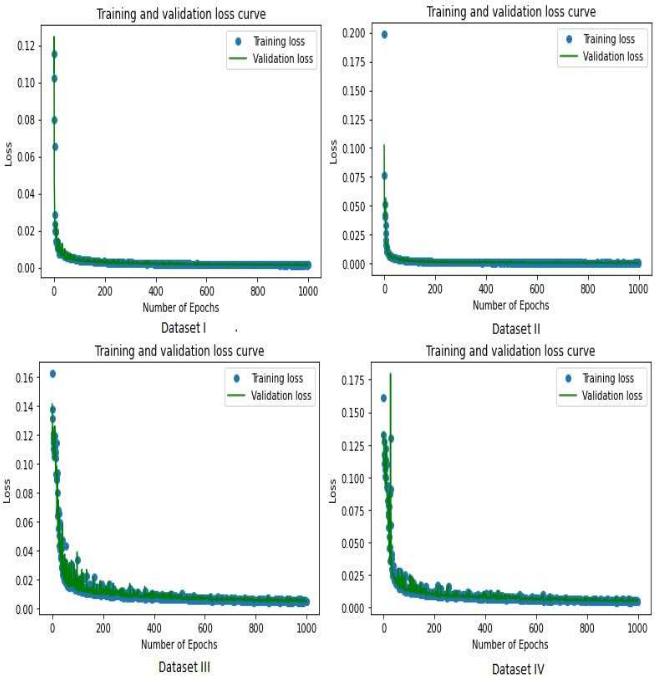


FIGURE 7. Training and validation loose curves for a CNN autoencoder throughout all four datasets of fingerprint images.

TABLE 3. Training Hyper-Parameters for CNN autoencoder

Parameter	Method
Number of Epochs	1000
Batch size	128
L2 regularization	0.005
Number of channels	1

V. EXPERIMENTAL RESULTS AND DISCUSSION

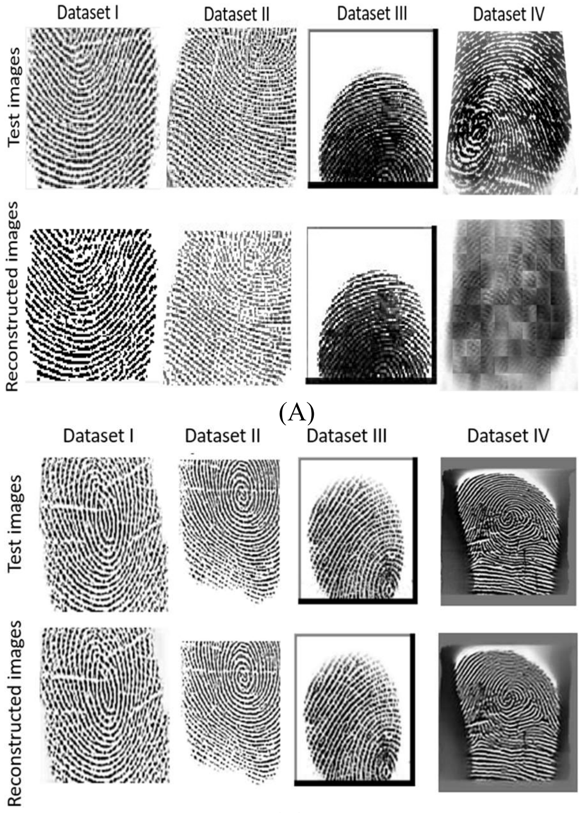
We demonstrated the predictions of fingerprint features on the four testing datasets with a sparse autoencoder, and the proposed CNN autoencoder. The variation for the performance achieved by the two investigated architectures were evaluated as the classification type and database varied. The mean square error (MSE) has been utilized to calculate the error between the estimated fingerprint features and the original fingerprint features. We used the MSE formula in Eq [6] because it is the most common estimator of image quality measurement metric. It is a full reference metric to calculate the differences in pixel values between the input and output images to evaluate the accuracy for autoencoders.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y})^2 \quad (6)$$

In Eq [6] MSE represents the mean square error, n is the number of data points, Y_i represents the observed value and $\hat{Y}$ represents the predicted value. According to the results from these experiments on the testing datasets, the mean square error (MSE) for sparse autoencoder has been improved by manual enhancement of the fingerprint images during the pre-processing activities. It showed that the process of cropping the images increased the learning capability for sparse autoencoder, allowing fast training time and enhanced the performance for the architecture. The CNN autoencoder achieved very good results for recreating the fingerprint features. The proposed algorithm showed better performance in comparison to the sparse autoencoder among four datasets, see Figure 8 A & B, and Table 4. It is observed that the features with complex patterns in the original fingerprint images produced the best latent representation in the recreated images by this model, which minimized the MSE error. Moreover, the proposed approach eliminated overfitting problems and any possible data leakage in the reconstructed images.

TABLE 4. Performance of the CNN autoencoder vs Sparse autoencoder among all datasets.

Model	MSE Dataset I	MSE Dataset II	MSE Dataset III	MSE Dataset IV
CNN autoencoder	0.0013	0.0015	0.0045	0.0048
Sparse autoencoder	0.023	0.026	0.033	0.07



(A)

(B)

FIGURE 8. The results for the reconstructed images by using (A) Sparse autoencoder (SAE), (B) CNN autoencoder on the testing sets for four datasets.

Further to our exploration, we performed the validation experiments on four testing datasets. This is to evaluate the fingerprint matching performance for the proposed model (CNN autoencoder) and sparse autoencoder among these datasets. We used cumulative match characteristics (CMC) as a performance evaluation between the reconstructed and the original features in the fingerprint images. Cumulative match characteristics is a metric used to assess the accuracy of algorithms that produce scores of possible matches in the biometric systems. Throughout the results, the proposed approach achieved very good identification rate with 98.1% for Dataset I, 97% for Dataset II, 95.9% for Dataset III, and 95.02% for Dataset IV, and overcomes sparse autoencoder identification accuracy, which was recorded at 92.3% for Dataset I, 90.01% for Dataset II, and 87.5% for Dataset III, and 70% for Dataset IV, see Figure 9 A & B.

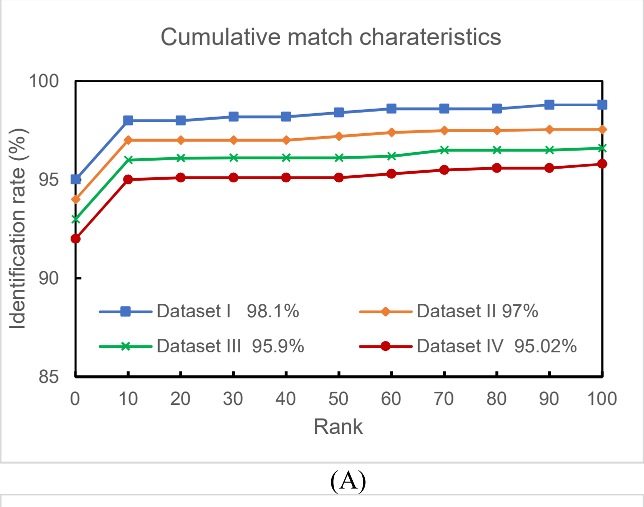


FIGURE 9. Cumulative match characteristics (CMC) curves display fingerprint matching results. (A) CMC results by the proposed approach (CNN autoencoder) (B) CMC results by Sparse autoencoder

This is to note that the proposed approach had better identification rate than sparse autoencoder on replicating the fingerprint features on Dataset IV which were collected from a fingerprint verification competition database (FVC2004). We compared the CNN autoencoder with other state-of-the-art methods [51], [52], which used the same fingerprint images from FVC2004 database. It can be seen that the proposed reconstruction algorithm produced the highest accuracy compared to these state-of-the-art algorithms, see Table 5. Therefore, the proposed approach has been performed effectively on real-word images vs. sparse autoencoder, and other methodologies. Hence, the proposed CNN model can be used for fingerprint identification in several application fields.

TABLE 5. Performance of the proposed approach vs state-of-the-art methods by using fingerprint images from FVC2004 database

Author	Algorithm	Accuracy
proposed approach	**CNN autoencoder**	**95.02%**
Gowthami et al [51]	Neural Network	94.24%
Gowthami et al [51]	Euclidean distance	92.79%
Deshpande et al [52]	CNNAI	80%

We measured the memory size for the proposed architecture, and sparse autoencoder, which are 1.257 MB. and 0.155 MB, respectively. This is the advantage of the proposed verses other pre-trained architectures such as SqueezeNet, Alexnet, Resnet50, and ShuffleNet. Figure 10 shows the comparison of the CNN autoencoder vs. the sparse autoencoder, and other pre-trained models in terms of memory size. These models utilize large CNN layers which require a massive disk size for deployment on an embedded system, and IoT devices.

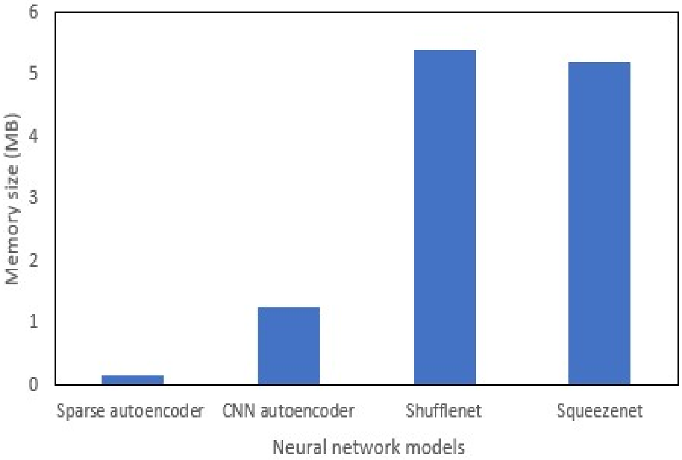


FIGURE 10. Comparison of the proposed approach (CNN autoencoder), and sparse autoencoder vs. other pre-trained models in terms of memory usage.

VI. CONCLUSIONS AND OTHER EXPERIMENTAL TARGETS

This work implements a novel approach for reconstructing fingerprint images based on CNN architecture. The CNN autoencoder is designed with an encoder and decoder, modelling the challenge of fingerprint image recreation by extracting the input image features, and replicating well-fined details in the output image. The proposed CNN autoencoder showed very good performance for replicating the fingerprint features from the images and overcomes the sparse autoencoder and other state of art methodologies in terms of calculating the mean square error between the estimated and the original features. Indeed, analyzing the obtained experimental results, the convolutional autoencoder is the most suitable technique for recreating complex context fingerprint features as it improved and sharpened the fingerprint features on real-world fingerprint images such as FVC2004 and SOCOFing databases. The measured memory size of the proposed CNN autoencoder is much lower than the state of art AI methods and this makes it suitable to run on low-cost embedded devices. Therefore, CNN autoencoder is viable option for biometric authentication and identification applications.

We foresee two different directions that can further enhance the performance of this model, one direction is to integrate the proposed approach within fingerprint scanners such as Lumidigm and Secugem sensors, which will give the final outcome on how the model performs in reconstructing the images. In addition to that, it could consider other data augmentation techniques using histogram-based operations and other geometric transformations to improve the mean square error value for the proposed approach.

ACKNOWLEDGMENT

We thank the Islamic Development Bank for the support of the Ph.D. work of A. Elhanashi and the “Dipartimenti di Eccellenza - Crosslab” project by MIUR.

REFERENCES

[1] Conti V, et al. A Multimodal Technique for an Embedded Fingerprint Recognizer in Mobile Payment Systems. Mobile Information Systems, 2014, 5 (2): 105-124.

[2] Alshehri H, et al. Alignment-Free Cross-Sensor Fingerprint Matching based on the Co-Occurrence of Ridge Orientations and Gabor-HoG Descriptor. IEEE Access, 2019, 7:86436-86452.

[3] Jin B, et al., Deep Facial Diagnosis: Deep Transfer Learning from Face Recognition to Facial Diagnosis. IEEE Access, 2020, 8: 123649-123661.

[4] Sang J., et al. An efficient fingerprint identification algorithm based on minutiae and invariant moment. Personal and Ubiquitous Computing, 22: 71–80, 2018.

[5] Figueiredo I., et al. Automated retina identification based on multiscale elastic registration. Computers in Biology & Medicine, 2016: 130-143.

[6] Zheng Q., Yang M., Yang J., Improvement of Generalization Ability of Deep CNN via Implicit Regularization in Two-stage Training Process. IEEE Access, 2018(6): 15844 - 15869.

[7] Luo C., et al. Gait Recognition as a Service for Unobtrusive User Identification in Smart Spaces. ACM Transactions on Internet of Things, 2020.

[8] Ezhilmaran D, Adhiyaman M. A review study on latent fingerprint recognition techniques. Journal of Information and Optimization Sciences, 2017, 38(3-4): 501-516.

[9] Dospinescu O, Lîsîi I. The recognition of fingerprints on mobile applications–an android case study. Journal of Eastern Europe Research in Business and Economics, 2016: 1-11.

[10] Shaheed K, Liu H, Yang G, et al. A systematic review of finger vein recognition techniques. Information, 2018, 9(9): 213.

[11] Yoon S., et al., Longitudinal study of fingerprint recognition. Proc. of the National Academy of Sciences, 2015, 112(28):8555.

[12] Deshpande U., et al. Automatic latent fingerprint identification system using scale and rotation invariant minutiae features. Int. Journal of Information Technology, 2020(1).

[13] Wang Tao, Zheng Zhigao, Kashif B, et al. FinPrivacy: A Privacy-Preserving Mechanism for Fingerprint Identification. ACM Transactions on Internet Technology (TOIT), 2020.

[14] Cao Kai Jain, et al. Automated Latent Fingerprint Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019, 41(4):788-800.

[15] Park C H, Lee J J, Smith M J T, et al. Directional filter bank-based fingerprint feature extraction and matching. IEEE Transactions on Circuits and Systems for Video Technology, 2004, 14(1): 74-85.

[16] Chen S., Zhou Y., Zhao B., et al. Orthogonal curved-line Gabor filter for fast fingerprint enhancement. Electronics Letters, 2014, 50(3):175-177.

[17] Chen T., et al. Fingerprint enhancement using the second-order oriented partial differential equation, IEEE CISP 2012.

[18] Chikkerur, S., et al., Fingerprint Image Enhancement Using STFT Analysis 2018. LNCS 3687, pp. 20–29, 2005

[19] Ghafoor M., et al., Fingerprint frequency normalisation and enhancement using two-dimensional short-time Fourier transform analysis. IET Computer Vision, 2017, 10(8):806-816.

[20] Bidishaw J., Nalini T., Two Stage Block-Wise Fingerprint Enhancement Using Discrete Wavelet Transform. Int. Journal of Computer Science & Information Technology 2014.

[21] Liu M, Liu S., Zhao Q. Fingerprint orientation field reconstruction by weighted discrete cosine transform. Information Sciences, 2014, 268:65-77.

[22] Ding S., et al. Combining Gabor filtering and classification dictionaries learning for fingerprint enhancement. IET Biometrics, 2017, 6(6):438-447.

[23] Zheng Q, Zhao P, Zhang D, and Wang H, MR-DCAE: Manifold regularization-based deep convolutional autoencoder for unauthorized broadcasting identification. International Journal of Intelligent Systems, 2021. DOI: 10.1002/int2.20210123

[24] Chakraborty S., Rao K., Fingerprint enhancement by directional filtering, IEEE Int. Conference on Electrical Engineering/electronics, Computer, Telec. And Inf. Tech. 2012.

[25] Jian Li, Jianjiang Feng, Jay Kuo C.-C., “Deep convolutional neural network for latent fingerprint enhancement,” Signal Processing: Image Communication, vol. 60, 2018, pp. 52–63.

[26] Manickam A. et al., “Score level based latent fingerprint enhancement and matching using SIFT feature,” Multimedia Tools and Applications, vol. 78, 2019, 3065.

[27] Cao, K., Nguyen, D.-L., Tymoszek, C. and Jain, A.K. End-to-End Latent Fingerprint Search. IEEE Transactions on Information Forensics and Security 15 (2020) 880-894.

[28] Wang Y., et al., Damaged fingerprint classification by Deep Learning with fuzzy feature points, IEEE CISP-BMEI 2016.

[29] Wang Y., et al. A robust damaged fingerprint identification algorithm based on deep learning, IEEE IMCEC 2017.

[30] Su H et al. A deep learning approach towards pore extraction for high-resolution fingerprint recognition. IEEE ICASSP 2017.

[31] Pandya B, et al. Fingerprint classification using a deep convolutional neural network, IEEE ICIM 2018.

[32] Li H. Feature extraction, recognition, and matching of damaged fingerprint: Application of deep learning network. Concurrency and Computation: Practice and Experience, 2020.

[33] Zia T, et al. Robust Fingerprint Classification with Bayesian Convolutional Networks. IET Image Processing, 2019, 13(8):1280-1288.

[34] Jeon W S, Rhee S Y. Fingerprint Pattern Classification Using Convolution Neural Network. International Journal of Fuzzy Logic & Intelligent Systems, 2017, 17.

[35] Peralta D, et al. On the use of convolutional neural networks for robust classification of multiple fingerprint captures. International Journal of Intelligent Systems, 2018, 33(1):213-230.

[36] Zeng F, Hu S, Xiao K. Research on partial fingerprint recognition algorithm based on deep learning. Neural Computing and Applications, 2018:1-10.

[37] Zheng Q, et al. Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification. Neural Computing and Applications, 2020: 1-23.

[38] Zheng Q, et al. CLMIP: cross-layer manifold invariance based pruning method of deep convolutional neural network for real-time road type recognition. Multidimensional Systems and Signal Processing, 2021, 32(1): 239-262.

[39] Ali A M, Farhan A K. A novel improvement with an effective expansion to enhance the MD5 hash function for verification of a secure e-document. IEEE Access, 2020, 8: 80290-80304.

[40] Ali A M, Farhan A K. Enhancement of QR code capacity by encrypted lossless compression technology for verification of secure E-Document. IEEE Access, 2020, 8: 27448-27458.
[41] Sun, Y., Wang, X., Tang, X., 2014. Deep learning face representation from predicting 10,000 classes, IEEE Conference on Computer Vision and Pattern Recognition, pp. 1891–1898.
[42] Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.
[43] Rumelhart, D.E., et al., Learning representations by back-propagating errors. Cognitive modeling 5, 1. 1988
[44] Garrison W. Cottrell, Paul Munro, and David Zipser. Learning internal representations from grayscale images: An example of extensional programming. Conference of the Cognitive Science Society, 1987.
[45] Hijazi, S., Kumar, R., Rowen, C., Using Convolutional Neural Networks for Image Recognition. 2015. https://ip.cadence.com/uploads/901/cnn_wp-pdf
[46] Maggipinto, M., et al., A Convolutional Autoencoder Approach for Feature Extraction in Virtual Metrology, Procedia Manuf., vol. 17, pp. 126–133, Jan. 2018.
[47] Ujwalla Gawande, et al., Fingerprint Color Image Database. v1 (https://www.mathworks.com/matlabcentral/fileexchange/52507-fingerprint-color-image-database-v1), MATLAB Central File Exchange. Retrieved January 8, 2021.
[48] Gawande U. et al., (2021). Fingerprint Grayscale Image Databasev2 (https://www.mathworks.com/matlabcentral/fileexchange/52508-fingerprint-grayscale-image-database-v2), MATLAB Central File Exchange. Retrieved January 8, 2021.
[49] Shehu, Y. et al., Sokoto Coventry Fingerprint Dataset. 2018, https://arxiv.org/abs/1807.10609
[50] Maio D. et al., "FVC2004: Third fingerprint verification competition," Int. Conf. on Biometric Authentication, Berlin, Heidelberg, pp. 1-7, 2004.
[51] Gowthami A. T., et al. (2015). Fingerprint Recognition Using Zone Based Linear Binary Patterns. Procedia Computer Science. 58. 552-557. 10.1016/j.procs.2015.08.072.
[52] Deshpande, U. et al., (2020). CNNAI: A Convolution Neural Network-Based Latent Fingerprint Matching Using the Combination of Nearest Neighbor Arrangement Indexing. Frontiers in Robotics and AI. 7. 113. 10.3389/frobt.2020.00113.