Scientific Research Publishing

Journal of Software Engineering and Applications, 2020, 13, 1-13
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

Reduction of False Rejection in an Authentication System by Fingerprint with Deep Neural Networks

Stéphane Kouamo[1], Claude Tangha[1], Olaf Kouamo[2]

[1]Department of Computer Science, University of Yaounde I, Yaounde, Cameroon
[2]Department of Mathematics and Statistics, Université Denis Didérot (Paris VII), Paris, France
Email: skouamo@gmail.com, ctangha@gmail.com, okouamo@yahoo.fr

How to cite this paper: Kouamo, S., Tangha, C. and Kouamo, O. (2020) Reduction of False Rejection in an Authentication System by Fingerprint with Deep Neural Networks. *Journal of Software Engineering and Applications*, **13**, 1-13.
https://doi.org/10.4236/jsea.2020.131001

Received: January 1, 2020
Accepted: January 28, 2020
Published: January 31, 2020

Abstract

Faultless authentication of individuals by fingerprints results in high false rejections rate for rigorously built systems. Indeed, the authors prefer that the system erroneously reject a pattern when it does not meet a number of predetermined correspondence criteria. In this work, after discussing existing techniques, we propose a new algorithm to reduce the false rejection rate during the authentication-using fingerprint. This algorithm extracts the minutiae of the fingerprint with their relative orientations and classifies them according to the different classes already established; then, make the correspondence between two templates by simple probabilities calculations from a deep neural network. The merging of these operations provides very promising results both on the *NIST*4 international data reference and on the *SOCFing* database.

Keywords

Authentication, Fingerprint, False Rejection, Neural Networks, Pattern Recognition, Deep Learning

1. Introduction

Fingerprints form a very specific class of models with singular particularity and proven statistical characteristics. Thus, the problems of fingerprint recognition seem to be much more constraining than other classical problems of form recognition (such as the recognition of manuscript characters) where neural networks have already been successfully applied [1] [2] [3]. Faultless authentication of individuals by fingerprints remains one of the main problems of fingerprint recognition. Indeed, the authors prefer that the system erroneously reject a pattern when it does not meet a number of predetermined correspondence criteria. This phenomenon tends to lead to an increase in false rejection rates when the fingerprint recognition system is designed for authentication [4] [5].

The main question we will address here is: how to design an algorithm that will reduce the rates of false rejections in a fingerprint authentication process while maintaining a minimum number of acceptable minutiae's?

The approach chosen for our work is to conduct a deep supervised learning from the neural network model. The system built for authentication will then be applied to data from an international *NIST*4 [6] database and the *SOCFing* database. Neural networks have proven their worth in many areas, including the pattern recognition [7] [8]. One of their particularities is the ability to adapt to the data to be processed and the ability to perform the calculations in parallel, allowing them to intervene in various fields of application.

We are working on a system for authentication, which is a *one-to-one* comparison. This kind of system tends to be more rigorous and, therefore, prefers to reject a pattern that does not have all the required criteria, even if the latter belongs to the correct class (which implies high false rejections). Our idea is to propose an algorithm that produces good results (compared to those existing), while at least avoiding rejecting a pattern of the correct class. This implies acting on the internal structure of the neural network and optimizing the recognition algorithm so that it rejects as little as possible a pattern erroneously.

Then, we propose a new algorithm for extracting and classifying minutiae by a parsimonious approach as well as the appropriate deep neural network structure with three hidden layers that use different activation functions at each looping during learning. This enabled us to improve the false rejection score (by 0.4 percentage on average), compared to existing ones.

The network we propose has an input layer, an output layer and a hidden layer consisting of 3 sub-layers (two sub-layers of classification and one sub-layer of correspondence whose size is strictly greater than that of the previous two). We opted for three (03) hidden layers due to the three major operations that underlie the authentication process, namely:

- The identification of the minutiae and their relative orientations;
- The constitution of the characteristic vectors according to the type of detail and the orientation of these;
- Finally, the correspondence of the templates constituted.

Each of these three operations will then be carried out at each of the hidden layers and the outputs/results of the upper layers will be used as inputs for the following layers, this in a hierarchical manner [9].

On the other hand, based on the theoretical basis of fingerprint authentication, it takes an average of 12 to 30 minutes to establish a match between two templates [10] [11]. This constraint also guided us in the choice of 03 hidden layers in the following way: the prototypes of the first hidden layer use the first 12 minutiae detected, those of the second hidden layer use the remaining minutiae and the whole is reconstituted at the corresponding layer (something that no technique had yet explored).

We also use a different activation function at each hidden layer and a leapfrog symplectic integrator for optimization of the result.

The rest of the document is organized into three sections. The first presents the main algorithms for fingerprint recognition. The second describes the proposed method and the third presents some of the results obtained. We'll end up with a conclusion.

2. Related Works

The henry classification reveals five major classes of fingerprint [12], this from the overall texture of the fingerprint image. We will focus more on the aspect of the correspondence which deals with a set of much more subtle elements to establish the match between two fingerprints namely (the minutiae's, the striations, their relative orientations, etc.). In fact, there are two main classes of algorithm processing a fingerprint using a deep learning to know: those that use locally connected neural networks and those that are based on probabilistic or convolutional algorithms, but let's present first the old classic method.

2.1. Classic Method

This method consists of comparing the pixel matrices of the images of two fingerprints, and calculating the correlation that exists between these pixels. The image to be recognized (M) as well as the images in the database are all scanned and recorded as a pixel matrix rated M_i. Recognition therefore consists in comparing the input matrix to all the M_i. matrices of the database for the case of identification, or to a single specific M matrix in the case of authentication [13] [14] [15].

There are also some classic methods that use minutiae's events to compare fingerprints. These minutiae's are extracted from two fingerprints, and represented as a set of points in the two-dimensional plane according to the coordinate model. The comparison is to find a good alignment of minutiae of two fingerprints (I and J) that produces a maximum of pairs of similar minutiae. This method uses an intermediate algorithm to extract main characteristics of fingerprint image, and simply compare formed vectors.

Limits

- Use of fingerprint image pixels that can vary significantly depending on the scan device, etc;
- No learning is required;
- The result depends on the algorithm of extraction of the main characteristics of fingerprint image.

2.2. Probabilistic Neural Network

Algorithm of this class of method use to deal exclusively with the aspect of correspondence, instead of also taking into account the aspect of classification [16] [17]. Here, operations consist of two main steps, namely: a treatment stage and a decision stage. The processing phase essentially aligns the two images and extracts from each one "*a central region*". Both central regions are used to feed the decision stage, which is the most appropriate part of the neural network of the algorithm and is subject to the formation of examples.

While the treatment phase is fairly standard (*i.e.*, uses existing treatment algorithms [16] [17]), the decision phase proposed by authors is quite innovative and is based on probabilistic neural networks that use a Bayesian approach to assess the probability that two *P* templates are identical. The network is formed by the descent of the gradient using a set of learning pairs of images from several different fingerprints. The addition of a few additional fingers to perform the phase of use consolidates the robustness of the neural network created and tested beforehand.

This technique uses a central image region with data alignment and compression; and then, calculation of probabilities to match two patterns provided as network input after multiple successive filters.

Limits

- Using complex formulas to calculate probabilities;
- Treatment and events on minutiae are associated with a single layer of the network;
- Usable for a limited number of images including the learning phase.

2.3. Locally Connected Neural Networks

In this algorithm, the process of recognition of a fingerprint consists of two important phases: the extraction of minutiae and the classification of the template [18].

- In the first step, an image is passed through the system input and is reduced to a vector of characteristic features, through various transformations, calculation of moments, etc.) and the prototype vectors are formed by database data class;
- In the second step, a direct classification or comparison that will lead to the recognition or not of the pattern is carried out between the characteristic input vector and the prototypes of the database. This locally connected neural network-based method is more appropriate for verification, security and identification applications.

The particularity of this technique is the use of the characteristic vectors that are invariant both to the translation and the rotation of the fingerprint image. Qian *et al.*, in their article [19], proposed a model of *CNN* that, when applied to fingerprint verification, through the Fingerprint Verification for Convolutional Neural Network (*FVCNN*) system, produces interesting, though perfectible results.

This technique is based on three classification algorithms: the middle classes, the nearest neighbour and the nearest neighbours [19] [20] [21].

The main advantage of this method is the use of prototypes (characteristic vectors of fingerprint images following predefined classes) invariants to translation/rotation to perform the recognition. This algorithm is appropriate for identifying an individual in a large database and uses the nearest neighbours to perform classification in a deep learning context.

Limits

- Using thresholds and activation calculations to match;
- Treatment and events on minutiae are associated with a single layer of the network;
- Learning times are high.

2.4. A Three Hidden Layers Neural Network

In order to reduce false rejection while maintaining a high recognition rate, we propose a new algorithm based on the use of deep neural network to perform learning and using stage, that contribute to reduce false rejection rate according to the existing methods. The structure of our neural network is then that of a locally connected neural network with three hidden layers, which uses the probability calculation to classify the sub-blocks of the image as input, taking into account Henry's classification. The first hidden layer is connected to the second hidden layer and is initialized with the prototypes of each class of data. Then the second classification layer is strictly connected to the correspondence layer, and the neurones of this correspondence are initialized with the whole data of the database. One of the particularities of the structure we propose is that at the hidden layers, the output of the upper level constitutes the entrances of the lower level and so on.

We choose two prototypes for each class of data (arch, whorl, left and right loop) according to the Henry's classification [22] and initialize the two first hidden layers with those prototypes, in order to facilitate the convergence of our algorithm. We also use a threshold (to determine if there is a match between two templates), a symplectic integrator and the basic retro propagation algorithm for the learning of the deep neural network. **Figure 1** shows the structure of the proposed deep neural network.

Where x_i is the input image vector while o_i is the expected output; h_i represents prototypes of each class of data, w_{ji} the connection weight between the input layer and the 1st classification layer; z_{ik} the connection weight between the two classification layer; g_k are prototypes of the second classification layer, z'_{kl} the connection weight between the 2nd classification layer and the correspondence layer, h'_l are the vectors of the entire database for the correspondence layer and w'_{li} the connection weight between the correspondence layer and the output layer.

Since it takes an average of 12 to 30 minutiae to establish the correspondence between two fingerprints, the structure of our network will be the following: the first hidden layer will be used to detect the first 12 minutiae, the second hidden layer will be used to detect the rest of the minutiae present which may be a

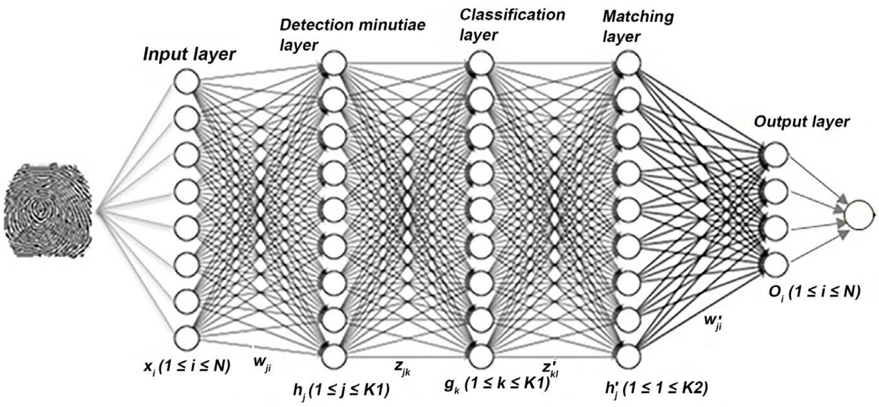

Figure 1. Structure of the proposed neural network.

number n (with $n \in [0, N]$, *N* being the total number of detectable minutiae of the fingerprint in question), and the third hidden layer is used to reconstruct all the minutiae, to detect and reconstruct the essential characteristics of the fingerprint.

Then the proposed algorithm can be applied for deep learning like this:

Algorithm: Fingerprint authentication and reduction of false rejection

1) Choose the initial prototypes two by two for each data class, according to the classification of Henry.

2) Initialize neurone of the first classification layer by the first prototype of each data class (according to the first 12 minutiae).

3) Initialize neurone of the second classification layer by the second prototype of each data class (according to the other remaining minutiae).

4) Initialize every correspondence layer neurone weight with the database pattern class by class.

5) Apply an input vector and then a dropout regularization.

6) Apply the data transformation on the detection minutiae layer (using function: *ReLU*, *Leaky ReLU*).

7) Apply the data transformation on 2^{nd} classification layer.

8) Apply the data transformation on the correspondence layer.

9) Select the winner.

10) Make comparison between predictions $\tilde{y}$ and the true target y, obtain loss function and loss score.

11) Optimize connection weights of each layer (using the l*eapfrog integrator*).

12) Return to (5) till finishing input vectors.

End Algorithm

We propose to use one of those three functions (*ReLU*, *Sigmoid* and *Leaky ReLU*) at each level of data transformation, and for one looping during the learning stage, all the three are used at different levels of the hidden layer to propagate the signal. For example, if at the first hidden layer for the i^{th} looping, we use the function *ReLU*, at the second hidden layer we will use Sigmoid and at the third hidden layer we will use *Leaky ReLU*.

Then, for $i^{th} + 1$ looping, we change the order of use of those functions through the hidden layers to compute the signal, and so on. This helps the network to improve system's capacities to adapt itself to a new pattern. The execution flowchart for this algorithm is proposed below by Figure 2.

By performing the training stage, the input vector to be recognized has therefore successively passed to the classification and matching layers. The prototypes of classification layers are used to make decision by probability calculations, between input and hidden layers; and the output at each level are used like to the input of the next level (see Figure 3).

3. Implementation and Results

To compute different training stage and test we use: a computer MacBook Pro *CPU* 2.5 GHz *Intel Core i5* with 6 *Go* 1600 MHz *DDR*3 memory; a computer Intel Core *i5 CPU* 2.5 GHz 2.5 GHz with 4 *Go* of RAM; a computer Intel Pentium *CPU G*645@ 2.90 GHz 2.90 GHz with 2 *Go* of *RAM* Operating system Ubuntu 12.10, Mac OS High Sierra and the application Octave 3.6.1, Jupyter notebook server 5.6.0, Python 3.70 and the library *Openmpi*.

The sampling is successively made by 40%, 50% and 80% of data from each database exactly as in [7] [8]. Then:

- For the *SOCOFing* database: 2400, 3000 and 4800 images also distributed between the individuals in the database;
- For the *NIST*4 database: 800, 1000 and 1600 images.

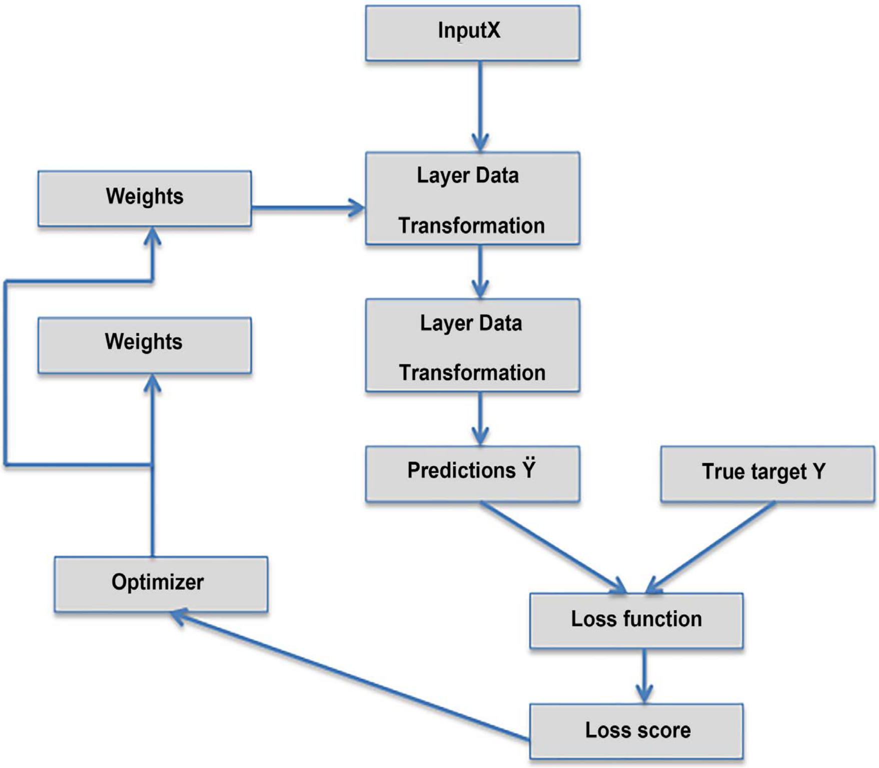

Figure 2. Execution flowchart of proposed method.

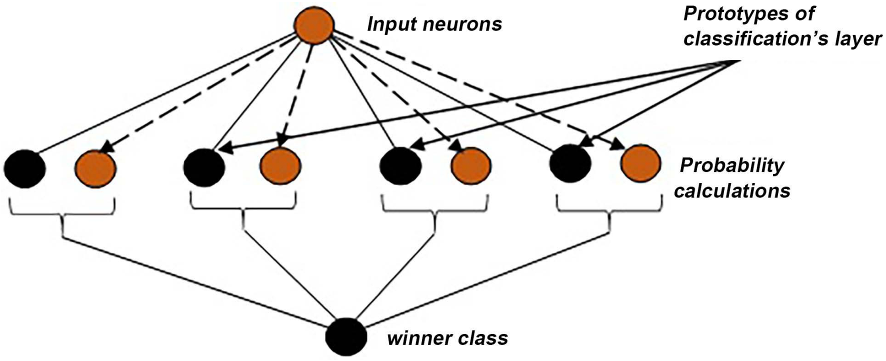

Figure 3. Probability calculation.

These images are proportionately distributed among the subclasses defined by Henry (*ark*, *tempted ark*, *vortex*, *and loops*).

The system performances will be evaluated by:

- The recognition rate (or Acceptation Rate) $AR = \frac{M}{N}$;
- The error rate constituted of false rejection rate (FRR) and false acceptation rate (FAR)

$$FRR = \frac{R}{N}; FAR = \frac{A}{N};$$

- The training and recognition time.

To ensure that our model does not over-learn, we use a "*Dropout*" regularization. This technique consists of disabling certain neurones at each looping so that the neurons are not dependent on each other (see Figure 4).

Thus, we place a "*dropout*" at the entrance of the hidden layers. It will be responsible for disabling 1% of the neurones (it is not recommended to go beyond 4% of neurones disabled by risk of facing under fitting).

Table 1 and Table 2 below show training results, while Table 3 and Table 4 show obtained testing results with both *NIST*4 and *SOCOFing* database.

3.1. Discussions

An analysis of the 2000 central images of the *NIST*4 database shows an average intensity of 102 with a standard deviation of 26. The minimum value of the main intensity is 21 and the maximum is197. On the other side, the analysis of the 6000 images in the S*OCOFing* database shows an average intensity of 254 with a standard deviation of 37. Gaussian noise with a standard deviation of $_{29}$ was added to each pixel of each sample in order to emulate the sound and later to increase the difference between samples of the same fingerprint.

The use of blocks size should be an arrangement depending on what the author finds. If we split fingerprint image into high blocks size we'll obtain a gain on training time but the performance of recognition rate will be not very good. In another way, if we use low block size, we'll have better performance on recognition rate but worst training time.

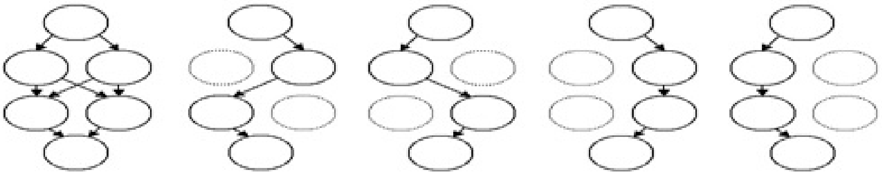

Figure 4. Dropout regularization.

Table 1. Training results with images sub-blocs sized of 8 × 8.

80% of NIST4	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	94.6	49.8	88.3	94.6	96.7	97.1
FRR (in %)	4.8	17.3	11.3	4.8	2.8	2.4
FAR (in %)	0.6	32.9	0.4	0.6	0.5	0.5
80% of *SOCOFing*	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	94.7	49.9	88.9	95.3	97.4	97.7
FRR (in %)	4.6	17.1	10.6	4.0	2.0	1.7
FAR (in %)	0.7	33.0	0.5	0.7	0.6	0.6

Table 2. Training results with images sub-blocs sized of 16 × 16.

80% of NIST4	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	93.8	49.1	87.6	94.0	96.1	96.5
FRR (in %)	5.5	17.9	11.9	5.3	3.3.	2.9
FAR (in %)	0.7	33.0	0.5	0.7	0.6	0.6
80% of *SOCOFing*	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	93.9	49.2	88.2	95.0	97.1	97.6
FRR (in %)	5.4	17.8	11.3	4.3	2.3	1.8
FAR (in %)	0.7	33.0	0.5	0.7	0.6	0.6

Table 3. Testing results with images sub-blocs sized of 8 × 8.

80% of NIST4	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	93.7	48.8	87.3	93.9	95.7	96.0
FRR (in %)	5.7	18.3	12.3	5.5	3.8	3.5
FAR (in %)	0.6	32.9	0.4	0.6	0.5	0.5
80% of *SOCOFing*	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	93.8	48.9	87.9	94.4	96.5	94.9
FRR (in %)	5.5	18.1	11.6	4.9	3.0	2.5
FAR (in %)	0.7	33.0	0.5	0.7	0.5	0.6

Table 4. Testing results with images sub-blocs sized of 16 × 16.

80% of NIST4	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	93.1	48.5	87.0	93.4	95.5	95.8
FRR (in %)	6.2	18.5	12.5	5.9	3.9	3.6
FAR (in %)	0.7	33.0	0.5	0.7	0.6	0.6
80% of *SOCOFing*	PM	LCNN	1 ppv LCNN	8 ppv LCNN	FVCNN deep *l* = 2	Proposed method
AR (in %)	93.2	48.9	87.8	94.1	96.1	96.5
FRR (in %)	6.1	18.1	11.7	5.2	3.3	2.9
FAR (in %)	0.7	33.0	0.5	0.7	0.6	0.6

The use of a *dropout* regularization helps the network to make good decision without under-learning and provide the ability to our proposed method to offer best results in terms of recognition rate and false rejection rate as shown by Table 3 and Table 4. We also use a particular technique to optimize our recognition result called *Tuning*. This technique consists on testing the neural network model by changing some parameter's values and at the end, to keep only those which provide better results. As we use different activation functions at each hidden layer to perform the training, associate with the fact that we permute them during all the stage, to give the ability to the network to adapt the right function with the particularities of data of a hidden layer, strongly contribute to reduce the proportion of error rate, especially false rejection rate as we expected. The *leapfrog* symplectic *integrator* [23] [24] [25] use and the input of hidden layers contribute to stabilizing our algorithm convergence.

In fact, our proposed algorithm performs matching in less than 2 s on average, on both databases used, with 18 minutiae. The overall system performs a false rejection rate by 2.7% on average against 3.2% on average for the other best techniques.

3.2. Proposition

In a context of deep learning, choose the number of hidden layers according to the number of major operations to be carried out in the field in question, and according to the characteristics underlying the problem to be solved, in order to optimize the choice of hidden layers (where the prediction itself is made) and to avoid introducing unnecessary layers. Each of these operations could then constitute a level of abstraction. These levels of abstraction would be treated individually, while considering the outputs/results of some as inputs of others and will avoid costly operations in time of calculating the "pooling".

Construction of a neural network (NR) according to the problem to be solved:

- Simple neural networks: use of several parameters to deal with the insufficient number of hidden layers. The quality of the result depends on the quality and number of parameters (which can grow exponentially when moving from a network of depths d to a network of depths $d + 1$).

- Deep neural networks: the quality of the result depends on local connections, convolutions and pooling. This can lead to too much learning time.
- Specific deep neural networks (our proposal): the quality of the result depends on the local connections, the operations inherent in the field of study and the underlying characteristics of the area. This makes it possible to concentrate on the essential and to avoid additional time-related costs and the complexity of the data to be processed.

Let's have:

- n : the number of major operations to solve a problem P;
- N: P's execution constraints;

If $n \leq N$, select a depth network n (hidden layer) to solve p and divide the constraints into n sub-parts.

4. Conclusions

The purpose of this paper was to propose a new algorithm allowing the authentication of an individual who wishes to access a computer system, using his fingerprints, in a context of reducing the false rejection rate. The obtained results show that our proposed algorithm, computed in a parallel way, perform matching in less than 2 s on average, on both databases used, with 18 minutiae. The overall system performs false rejection rate by 2.7% on average against 3.2% on average for the other technique. In fact, the proposed method is quite innovative and tries to combine the particularity of each data of the fingerprint's image and the way they are linked together when performing training stage. On the other hand, we introduce a dropout regularization to deal better with the over-learning.

One of the strong result of our method is based on the fact that, we made a special effort to select at each level layer the best activation function by introducing a supplementary parameter which helped us to use the right activation function at the right hidden layer, and then choosing the right winner neuron of the winner class at each level; this gives the capacity to our data to become invariants to translation and rotation. We also use a threshold (10^{-7} with 18 minutiae) to establish the matching between the expected output and that obtained.

One of the weak points of the technique we propose is that we need a very high number of data sampling to train the network. Lack of those data will not give the capacity to the network to handle latent images very well. One way to solve this problem consists on making pre-learning of some layers or making pooling during the training. But these techniques are risky, because the network could be better to provide good results only on those data.

5. Perspectives

Future work could focus on developing an authentication system that uses in addition to fingerprints and other features such as the way how the finger is placed on the sensor and some face characteristics [26]. These particular features will provide the system the ability to analyze a set of other elements specific to the individual behavior of each user and thus making the system able to identify a sudden change of behavior of that user. Other future works could also look at the choices and the number of times that *dropout* operations must be applied, in a context of deep learning with comparison of characteristic vectors invariants to translation/rotation by probability calculations. Questions about how many characteristics to switch to a generative adversarial network to produce the right fingerprint image could also be another line of research.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Jiang, J. (1998) Image Compression with Neural Networks: A Survey. *Signal Processing: Image Communication*, **14**, 737-760. https://doi.org/10.1016/S0923-5965(98)00041-1

[2] Kouamo, S. and Tangha, C. (2012) Handwritten Character Recognition with Artificial Neural Network. In: *Distributed Computing and Artificial Intelligence*, Advances in Intelligent and Soft Computing, Vol. 151, Springer Verlag, Berlin, 535-543. https://doi.org/10.1007/978-3-642-28765-7_64

[3] Kouamo, S. and Tangha, C. (2016) Fingerprint Recognition with Artificial Neural Networks: Application to E-Learning. *Journal of Intelligent Learning Systems and Applications*, **8**, 39-49.

[4] Viger, F.P., Nandar, K.W., Li, K. and Chen, J. (2019) Fingerprint Classification and Identification Algorithms for Criminal Investigation: A Survey. *Future Generation Computer Systems*.

[5] Rupali, S.P., Sonali, D.P. and Sudeep, D.T. (2018) Performance Evaluation of Fingerprint Trait Authentication System. Advances in Intelligent Systems and Computing, Vol. 632, Springer-Verlag, Berlin, 143-151. https://doi.org/10.1007/978-981-10-5520-1_14

[6] Watson, C.I. and Wilson, C.L. (1992) NIST Special Database 4 Fingerprint Database. National Institute of Standards, Technology, Advanced Systems Division, Image Recognition Group.

[7] LeCun, Y., Chopra, S., Hadsell, R., Marc'Aurelio, R. and Huang, F. (2006) A Tutorial on Energy-Based Learning. In: Bakir, G., Hofman, T., Schlkopf, B., Smola, A. and Taskar, B., Eds., *Predicting Structured Data*, MIT Press, Cambridge, 10-21.

[8] Maio, D. and Maltoni, D. (1998) Neural Network Based Minutiae Filtering in Fingerprints.

[9] LeCun, Y., Bengio, Y. and Hinton, G. (2015) Deep Learning. *Nature*, **521**, 436-444. https://doi.org/10.1038/nature14539

[10] Bengio, Y. and Delalleau, O. (2011) Shallow vs. Deep Sum-Product Networks. *Neural Information Processing Systems*, Sierra Nevada, 16-17 December 2011, 1. https://doi.org/10.1007/978-3-642-24477-3_1

[11] Capelli, R., Lumini, A., Maio, D. and Maltoni, D. (2002) Synthetic Fingerprint-Database Generation. *Proceeding of the* 16*th International Conference on Pattern Recognition*, Quebec City, August 2002, Vol. 3, 744-747.

[12] International Biometric Group (2011) The Henry Classification. http://www.biometricgroup.com

[13] Kouamo, S. and Tangha, C. (2013) Images Compression with Artificial Neural Network. Advances in Intelligent and Systems and Computing, Vol. 189, Springer Verlag, Berlin, 515-524. https://doi.org/10.1007/978-3-642-33018-6_53

[14] Kouamo, O. and Gouy-Pailler, C. (2013) Multi-Scale Test Procedure for Non-Stationarity in Short and Long Memory Time Series. *IEEE*, *ICASSP*, Vancouver, 26-30 May 2013, 5368-5372. https://doi.org/10.1109/ICASSP.2013.6638688

[15] Maltoni, D., Maio, D., Jain, A.K. and Prabhakar, S. (2003) Handbook of Fingerprint Recognition. Springer, New York.

[16] Baldi, P. and Chauvin, Y. (1993) Neural Networks for Fingerprint Recognition. *Neural Computation*, **5**, 402-418. https://doi.org/10.1162/neco.1993.5.3.402

[17] Specht, D.F. (1990) Probabilistic Neural Networks. *Neural Networks*, **3**, 109-118. https://doi.org/10.1016/0893-6080(90)90049-Q

[18] Thomas, T.J. (2000) Locally-Connected Neural Network for Fingerprint Recognition. *Proceedings of the IASTED International Conference*, *Intelligent Systems and Control*, Honolulu, 2000, 431-441.

[19] Qian, Y., Dong, J., Wang, W. and Tan, T. (2015) Deep Learning for Steganalysis via Convolutional Neural Networks. *Media Watermarking*, *Security*, *and Forensics*, Vol. 9409, 1-10. https://doi.org/10.1117/12.2083479

[20] Jagtap, V.N. and Mishra, S.K. (2014) Fast Efficient Artificial Neural Network for Handwritten Digit Recognition. *International Journal of Computer Science and Information Technologies*, **5**, 2302-2306.

[21] Yunsick, S. (2016) Intelligent Security IT System for Detecting Intruders Based on Received Signal Strength Indicators. *Entropy*, **18**, 366. https://doi.org/10.3390/e18100366

[22] Galton, F. (1892) Fingerprint. McMillan, London.

[23] Simo, J.C. and Tarnow, N. (1994) A New Energy and Momentum Conserving Algorithm for the Non-Linear Dynamics of Shells. *International Journal for Numerical Methods in Engineering*, **37**, 2527-2549. https://doi.org/10.1002/nme.1620371503

[24] Skeel, R.D. (1998) Integration Schemes for Molecular Dynamics and Related Applications. Department of Computer Science (and Beckman Institute), University of Illinois, Urbana.

[25] Yoshida, H. (2001) Non-Existence of the Modified First Integral by Symplectic Integration Methods. *Physics Letters A*, **282**, 276-283. https://doi.org/10.1016/S0375-9601(01)00186-4

[26] Hamayun, A.K. (2017) Feature Fusion and Classifier Ensemble Technique for Robust Face Recognition. *Signal Processing*: *An International Journal*, **11**, 1-15.