American Journal of Biomedical Engineering 2013, 3(2): 31-40
DOI: 10.5923/j.ajbe.20130302.01

A Fingerprint Verification Tool Using Adaptive Resonance Theory Nets

Tirtharaj Dash[1], Tanistha Nayak[1], Subhagata Chattopadhyay[2,*], Fethi A. Rabhi[3]

[1]Dept. of Computer Science and Engineering, Veer Surendra Sai University of Technology Burla, Odisha 768018, India
[2]Dept. of Computer Science and Engineering, Camellia Institute of Engineering, Kolkata 700129 West Bengal, India
[3]School of Computer Science, The University of New South Wales, Sydney 2015, Australia

Abstract Accurate verification of fingerprints is important to prevent hassles related to one's identification. The hassle could be at the recognition of one's own as well as the forged prints of others. The objective of this work is to develop a fingerprint verification tool using JAVA. The algorithms of Adaptive Resonance Theory (ART) net – ART-1 and ART-2 have been used. These algorithms have been implemented with 'C' language. Two hundred and twenty two genuine versions of finger prints have been used as training cases and 518 very similar looking but forged fingerprints have been used as test cases. Here, the optimum vigilance parameter (ρ) is obtained through carefully conducted parametric studies. Finally, a flexible error threshold has been selected to accept fingerprints with 95% matching in the pixel patterns and the rest are rejected. The study observes that, ART 1 and 2 are able to identify forged fingerprints with Total Success Rate (TSR) of 95.80% and 97.37%, respectively.

Keywords Biometrics, Fingerprints, Forgery, Verification, Adaptive Resonance Theory Net, Graphic User Interface

1. Introduction

Biometrics is the secure, reliable, and convenient methods to verify the identity of a person. It has now largely replaced the traditional paper-based (e.g., signature-based documents), password-protected devices (e.g., computers), and identity card-based systems (e.g., employee authorization). One important advantage of the biometrics is that, they are devoid of any worry of losing the information by losing the identity card or paper or forgetting the passwords[1][2]. Generally, left thumb impression (LTI) is considered to be most commonly used biometry in many cases[3]. The issues with LTI are that these could be distorted (e.g., in heavy duty manual labours) and therefore become inconclusive, when viewed by the naked eyes. As a serious consequence, such issues open the scope for forgeries. Given this practical scenario, this paper focuses on the development of a tool-based detection of an LTI using Adaptive Resonance Theory (ART) Nets – ART-1 and ART-2 those use the concept of Artificial Neural Network (ANN).

Neural Networks (NN) learn and adapt exemplary patterns by observations[4] and has been successfully applied to process very complex and large datasets[5][6][7]. There can be of two types of learning modes, such as supervised and unsupervised learning. Adaptive Linear Net (ADALIN), Multiple ADALIN (MADALIN), Perceptron Network, etc. are some examples of supervised learning methods. These types of NNs learn faster and more accurately, but the problem with these is that, new trainings are required each time they learn new input patterns and as a result, the previously learned patterns are lost. On the other hand, networks, such as Counter Propagation Network (CPN), Adaptive Resonance Theory Net (ART), and Kohonen's Self Organizing Map (SOM) rely on unsupervised learning and can store previously learned patterns well. Among these networks, ARTs in particular, are able to store new patterns without losing the memory of older patterns and thus advantageous over the supervised methods[8]. It is worth noting that Associative Memory Net (AMN) could be another useful technique that retains the patterns, learned in the past[9].

The key focus of this paper is to develop a faster Graphical User Interface (GUI) using ART-1 and 2 algorithms for fingerprint verification, offline. In this GUI-based system, the developed algorithms will be working as the backend of the tool, which were implemented using 'C' language. To train the algorithms a large database of genuine fingerprint patterns have been used. The advantage of using ART is its ability to capture the complexity of the fingerprint patterns without losing the previously learned patterns. The verification task is performed offline by computing a mismatch value. The front end of the tool has been developed with JAVA. The concept of native programming has been used to set a communication between 'C' and JAVA. As there are two algorithms, such as ART-1 and

* Corresponding author:
subhagatachatterjee@yahoo.com (Subhagata Chattopadhyay)
Published online at http://journal.sapub.org/ajbe

ART-2, these have been compared according to their performances.

Rest of the paper is organized as follows: section 2 studies related literature; proposed methodology is discussed in section 3; results and discussions are given in section 4; and the work is concluded and future scopes are discussed in Section 5.

2. A Brief Overview of Previous Works

Different techniques are being implemented for fingerprint detection. Below we have showcased few interesting studies in past ten years.

Yang and Park, (2008) proposed a fingerprint verification algorithm with tessellated invariant moment features[10]. This method could be applied to poor quality images of the fingerprints also. Authors had used Eigen value-weighted cosine distance for finding similarity between images. The algorithm yielded high detection accuracy. Dash et al., (2012) studied three algorithms, such as Associative Memory Net (AMN), ART-1 and ART-2 in the detection of 'forged' handwritten signatures and compared their respective performances[11]. The authors used OpenMP for developing and implementing the algorithms. The study observes that ART-1 and ART-2 were able to detect forged signatures with high accuracy (98.99% and 99.99%) compared to AMN (78.68%). Li et al., (2012) developed a cryptosystem by combining fingerprints with error correction codes and the work showed an improved performance[12]. Global and local structures of a fingerprint could be used by (Mali and Bhattacharya, 2011) for fingerprint verification[13]. However, in this work, simple matching technique had been used for authentication purpose, which might not be successful in the verification of distorted images or fingerprints taken with some angular deviation. A fusion system using biometrics like fingerprints, face, retina etc. was developed by Ross and Jain, (2003)[14]. This method addressed the information fusion in biometrics. This work was similar to work of Prabhakar and Jain, (2002) where the authors addressed decision-level fusion[15]. A fingerprint retrieval system was developed based on level-1 and level-2 features of the sample where level-1 refers to local orientation or frequency and level-2 is minutiae by Cappelli and Ferrara (2012)[16]. The error in the detection was only 1-2%. A mathematical model based fingerprint verification system was proposed by Jin et al., (2004)[17]. The model was developed on Integrated Wavelet and Fourier-Mellin transforms. The equal error rate was 1.01% in this work. A state of art research was conducted by Maio et al., (2002) in Fingerprint verification competition where total four algorithms were tested on a large number of data sets[18]. A fingerprint verification system was developed by Bazen et al., (2000) based on the principle of signal correlation[19]. In this work, template matching was used in the secondary stage of the verification. The equal error rate (EER) in their work was found to be 7.98%[19]. Two algorithms from top ten presented in fingerprint verification competition (FVC) in the year 2004 took an average of 3.5 seconds[20] to check the forgery. So in this case, our algorithms proposed in the next section achieve better result in minimal time consumption. Triangular method of matching of fingerprint pattern could be used by KovaÂcs-Vajna (2000) to develop a verification system[21]. Intermediate phases worked with minutiae extraction and dynamic time wrapping (DTW). The correct verification was found to be 85%.

3. Proposed Methodology

Methodology includes steps as described below. Fig.1 shows the outline of the proposed methodology, followed by a discussion of each step.

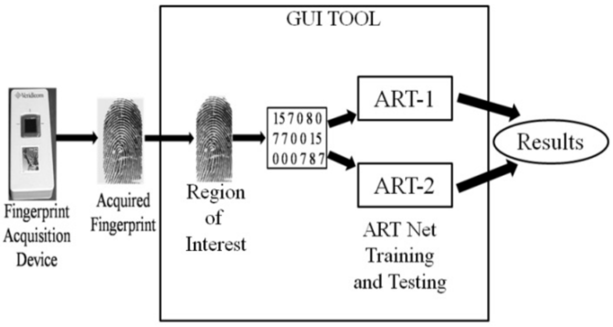
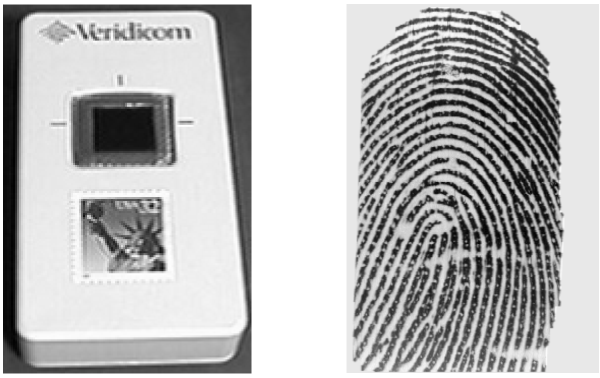


Figure 1. Outline of Proposed Methodology

3.1. Acquisition of Fingerprint

A compact solid state sensing device was used for acquisition of fingerprints from different persons. The device is given in Fig.2(a) below and the acquired fingerprint from the device is shown in Fig.2(b).



(a) (b)

Figure 2. **(a)** Fingerprint Acquisition Device, **(b)** fingerprint by the device

A total of **740** fingerprints were collected from different persons over duration of about **25** days. The whole fingerprint database was then divided into two classes namely, (i) genuine class (ii) imposter class.

1. Number of Genuine Class : **222**
2. Number of Imposter Class: **518**

Then the similarity index (SI) was calculated between genuine class fingerprints and corresponding imposter class fingerprints. The SI was calculated using a formula given in equation-1.

$$SI = \frac{1 - D_p}{T_p} \times 100 \qquad (1)$$

The arrangements of pixels in the forged fingerprints are compared with that of the original fingerprint, row-column-wise. The disparities are then computed. In equation 1, D_p is the number of 'dissimilar pixels' and T_p is the total number of pixels.

3.2. Formation of Region of Interest (ROI)

Region of interest was found out based on change in pixel value of the fingerprint image. A typical figure is given in Fig.3 to demonstrate the methodology of selecting the region of interest.

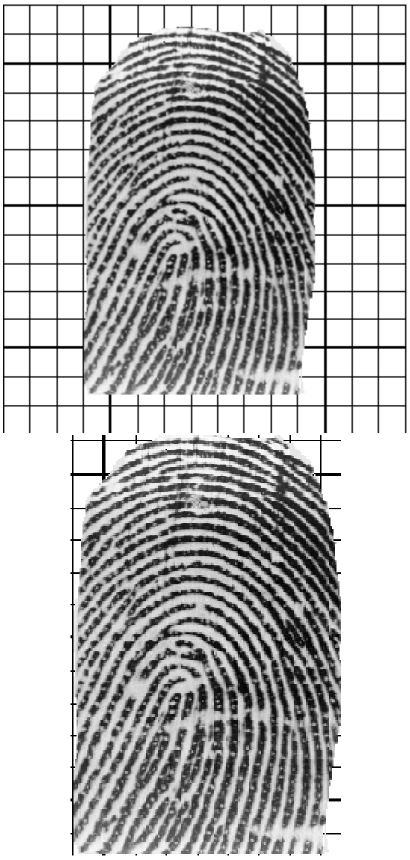


Figure 3. Fingerprint and the corresponding region of interest

3.3. Feature Extraction from the ROI

It is wise to mention that this work proposes two algorithms based on which the tool has been developed using JAVA. One algorithm uses the concept of ART net of Type-1 and other one uses the concept of ART net of Type-2. ART-1 uses the binary training and testing database for its operation whereas the later type of net uses continuous decimal values. So, the feature (pixel) extraction followed the formula given in equation-2.

$$\text{Gray Val} = 33\%\ \text{Red pixel} + 56\%\ \text{Green pixel} + 11\%\ \text{Blue pixel} \qquad (2)$$

This gray value was used for ART-2 network; whereas, for ART-1 net following conversion was done.

```
if (Gray Val)
        flag bit = 1;
else
        flag bit = 0;
```

3.4. Proposed Algorithm

The proposed algorithm for this work is given in a pseudo-code below. It should be noted that, the algorithm was implemented in 'C' language whereas the GUI tool was developed using JAVA. Both the program (C and JAVA) were linked using Native programming (JAVA Native Interface)[19]. Reason for such implementation is discussed in later section.

```
START
Initialize learning rate ($\alpha$), vigilance parameter ($\rho$), initial weights ($b_{ij}(0)$, $t_{ji}(0)$)
    /*Total number of iteration = Number of training patterns*/
WHILE (Stopping condition is FALSE)
  DO
      FOR each input training fingerprints
      DO
          Get the pixels of the genuine fingerprints
          $F_1$-Layer Processing
IF (Reset is TRUE)
Find the victim unit ($F_2$ unit) to learn the current input Pattern
Calculate $F_1(b)$ unit from $F_1(a)$ and $F_2$
ELSE
              Perform weight updation
      END
  END
REPEAT WHILE loop for the tested imposter fingerprints
STOP
```

Updation of weights is performed using the following mathematical equations:

Equations 3.1 and 3.2 correspond to the bottom-up and top-down weight updation respectively in ART-1.

$$b_{ij}(new) = \frac{\alpha x_i}{\alpha - 1 + \|x\|} \qquad (3.1)$$

$$t_{ji}(new) = x_i \qquad (3.2)$$

Similarly, Equations 4.1, 4.2 refers the weights of ART-2 net.

$$b_{iJ}(new) = \alpha du_i + \{[1 + \alpha d(d-1)]\} b_{iJ} \qquad (4.1)$$

$$t_{Ji}(new) = \alpha du_i + \{[1 + \alpha d(d-1)]\} t_{Ji} \qquad (4.2)$$

The symbols used in the above equations are described below.

$b_{iJ}(new)$= Updated bottom-up weight of winner node J in F_2 layer

$t_{Ji}(new)$= Updated top-down weight of winner node J in F_2 layer

α=learning rate

$\|x\|$=norm of vector x and is defined as in equation 3.1 (ART-1), 3.2 (ART-2).

$$\|x\| = \sum_{i=1}^{n} x_i \qquad (5.1)$$

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2} \qquad (5.2)$$

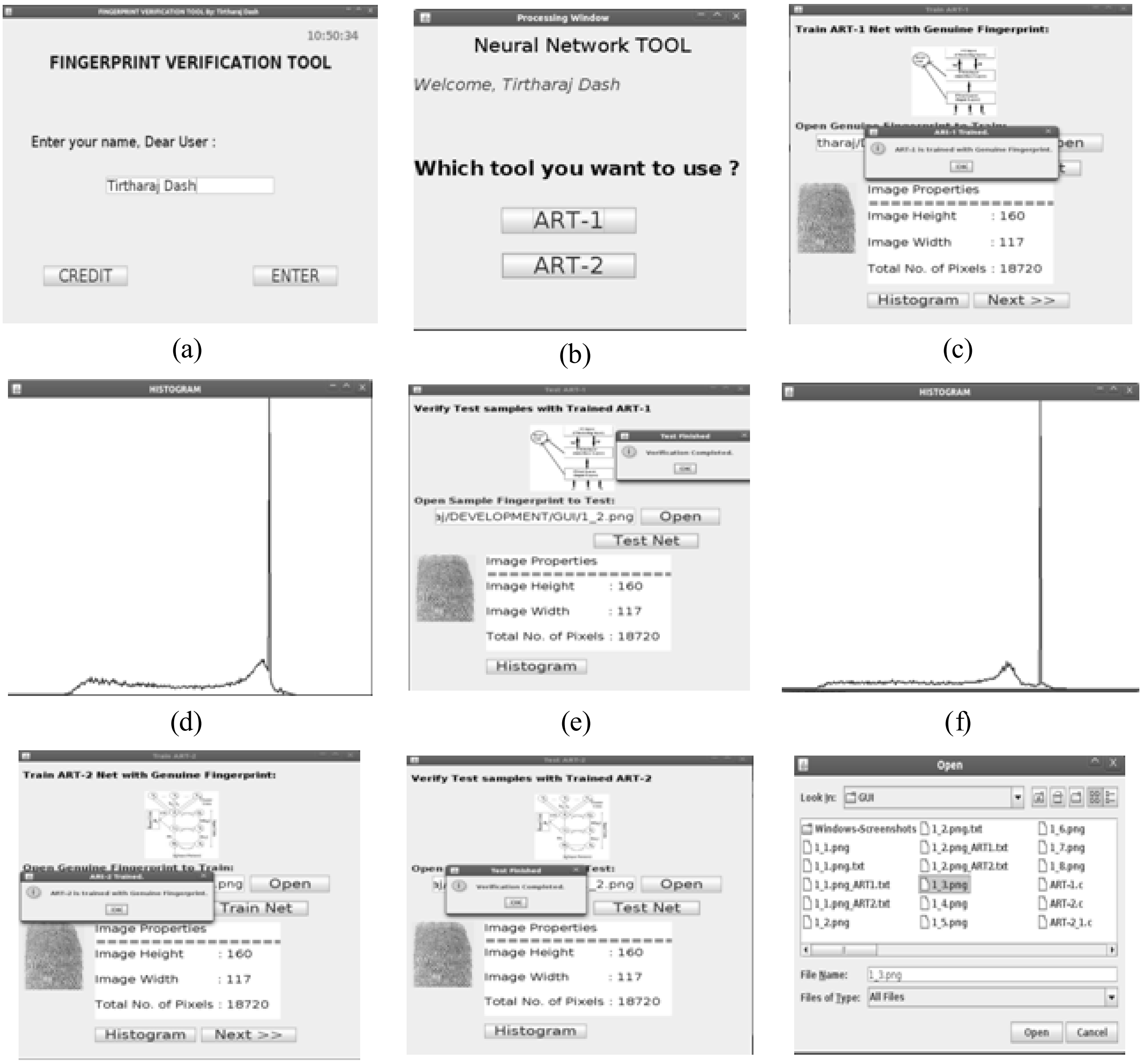
After ART net gets trained with all the training inputs (Genuine class fingerprints), the mismatch for the test case (imposter class fingerprint) can be calculated using equation 6.

$$mismatch = \left[1 - \frac{b^{*}_{ij}}{count}\right] \times 100 \qquad (6)$$

In this equation, '*count*' denotes the total number of bottom-up weights (b_{ij}) and 'b^*_{ij}' are the weights which are matched with that of the genuine class training fingerprints. Based on the mismatch value, the imposter class could be accepted or rejected. It is important to note that mismatch acts as a threshold and could be user/application specific.

3.5. Developed GUI and Running of the Tool

Below figures/windows (Fig.4a through j) show the GUI developed and its usage in this research work.



FINGERPRINT VERIFICATION TOOL By: Tirtharaj Dash

10:50:34

FINGERPRINT VERIFICATION TOOL

Enter your name, Dear User : Tirtharaj Dash

(a)

Processing Window

Neural Network TOOL

Welcome, Tirtharaj Dash

Which tool you want to use ?

(b)

Train ART-1

Train ART-1 Net with Genuine Fingerprint:

Open Genuine Fingerprint to Train: tharaj/[illegible]

ART-1 Trained.

ART-1 is trained with Genuine Fingerprint.

Image Properties
==================
Image Height : 160
Image Width : 117
Total No. of Pixels : 18720

(c)

HISTOGRAM

(d)

Test ART-1

Verify Test samples with Trained ART-1

Test Finished

Verification Completed.

Open Sample Fingerprint to Test: aj/DEVELOPMENT/GUI/1_2.png

Image Properties
==================
Image Height : 160
Image Width : 117
Total No. of Pixels : 18720

(e)

HISTOGRAM

(f)

Train ART-2

Train ART-2 Net with Genuine Fingerprint:

Open Genuine Fingerprint to Train: .png

ART-2 Trained.

ART-2 is trained with Genuine Fingerprint.

Image Properties
==================
Image Height : 160
Image Width : 117
Total No. of Pixels : 18720

(g)

Test ART-2

Verify Test samples with Trained ART-2

Open [illegible] Test: aj/[illegible]_2.png

Test Finished

Verification Completed.

Image Properties
==================
Image Height : 160
Image Width : 117
Total No. of Pixels : 18720

(h)

Open

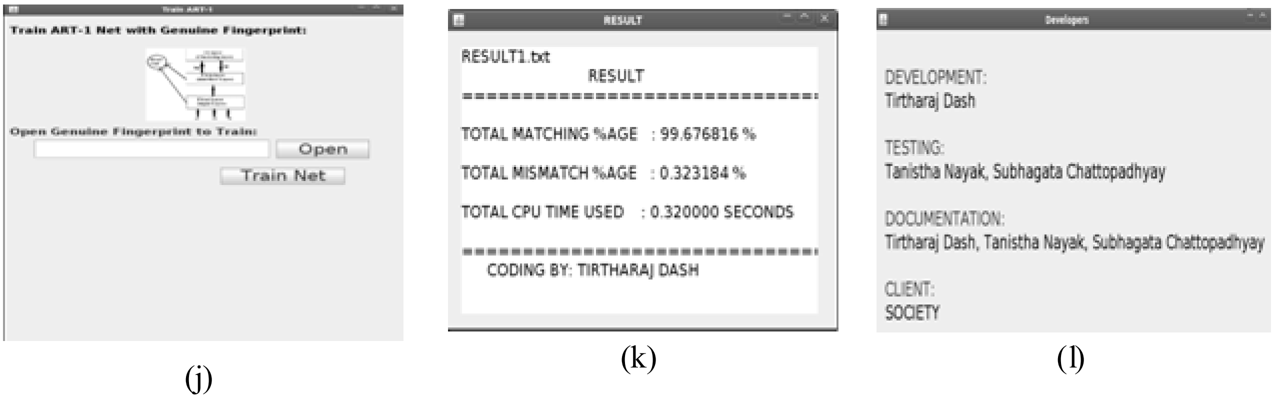
Look In: GUI

Windows-Screenshots | 1_2.png.txt | 1_6.png
1_1.png | 1_2.png_ART1.txt | 1_7.png
1_1.png.txt | 1_2.png_ART2.txt | 1_8.png
1_1.png_ART1.txt | 1_3.png | ART-1.c
1_1.png_ART2.txt | 1_4.png | ART-2.c
1_2.png | 1_5.png | ART-2_1.c

File Name: 1_3.png

Files of Type: All Files

(i)



Train ART-1 Net with Genuine Fingerprint:

Open Genuine Fingerprint to Train:

(j)

```
RESULT1.txt
              RESULT
================================

TOTAL MATCHING %AGE   : 99.676816 %

TOTAL MISMATCH %AGE   : 0.323184 %

TOTAL CPU TIME USED   : 0.320000 SECONDS

================================
    CODING BY: TIRTHARAJ DASH
```

(k)

DEVELOPMENT:
Tirtharaj Dash

TESTING:
Tanistha Nayak, Subhagata Chattopadhyay

DOCUMENTATION:
Tirtharaj Dash, Tanistha Nayak, Subhagata Chattopadhyay

CLIENT:
SOCIETY

(l)

Figure 4. The GUI and its running windows: **(a)** Welcome window of the tool **(b)** User directed to Tool choice window on pressing 'Enter' button **(c)** ART-1 Trained with one the genuine fingerprint and showing the property of the image **(d)** the histogram of the image can be viewed on clicking 'Histogram' button **(e)** ART-1 tested for the imposter pattern showing testing completion dialog **(f)** showing histogram of tested imposter fingerprint **(g-h)** Training and Testing of ART-2 network **(i-j)** fingerprint file browsing window and input for training in case of the fingerprint auto-detection failure by the tool **(k)** showing test result **(l)** showing persons of credit

3.6. Running of the Tool

When the JAVA tool is executed in JAVA environment, a welcome window appears and asks the user for name. The purpose of creating this window is to save the result data against the current user's name for future references and uses. As, this tool has been made for testing purpose only, it ask about the tool choice. That is, whether to choose ART-1 or ART-2 net. After choosing one tool, the user scans its left thumb in the acquisition device and click 'train network' button. If the user has already stored its fingerprint image in the computer, then no need to scan again. He/she can directly upload it into the tool using 'open' button. The tool then shows the image properties and the corresponding histogram. The next step is to test the network with imposter patterns. For this purpose, the same procedure is followed as stated above. Finally the result window opens and shows the matching percentage and computing time used and the data is appended in the user name file.

4. Experimental Results and Analysis

The contribution of this work is an applicable GUI tool for fingerprint verification and a detailed parametric study on the vigilance parameter (ρ) using the principles of ART-1 and 2. Then the CPU time has been computed to note the decision making speed by the tool. The study is carried out on the created fingerprint database. This work also compares performances of both ART-1 and ART-2 based algorithms for verification purpose.

Basically, performance of a biometric verification algorithm refers to its error rate and accuracy tested under many possible conditions. Four important terms have been used in this section for this purpose. Those are False Acceptance Rate (FAR), False Rejection Rate (FRR), Total Success Rate (TSR) and Equal Error Rate (ERR) and are given in equations 7-10, below.

- $$\text{FAR} = \frac{N_a}{N_i} \times 100\% \quad (7)$$
- $$\text{FRR} = \frac{N_r}{N_g} \times 100\% \quad (8)$$
- $$\text{TSR} = \left[1 - \frac{FAR + FRR}{T}\right] \times 100\% \quad (9)$$
- $$\text{EER} = \frac{FAR + FRR}{2} \quad (10)$$

Where,
N_a = Number of accepted **imposter** class patterns;
N_r = Number of Rejected **Genuine** class patterns;
N_i = Total number of imposter class patterns;
N_g = Total number of genuine class patterns;
T = Total number of attempts for test

4.1. Results of ART Nets

Table-1 and 2 shows FAR and FRR achieved by the developed ART-1 and 2 algorithms, respectively. The algorithm has been tested with various values of 'ρ' ranging from 0.50 to 0.99 as 0<ρ<1. It should be noted that for decision making, we have set the threshold as low as 5% mismatch to count number of accepted or rejected fingerprint patterns from the created database of 740 patterns. Hence, any mismatch threshold <5% is considered as 'accepted' and vice versa. By setting a stricter mismatch threshold, such as <1%, these false acceptances could be averted. It is wise to mention that the number of cluster unit (m) in ART nets is set to 20.

It should be noted that the average computation time for each vigilance parameter was calculated as the average of ten executions. However, time required to train the network with whole 222 fingerprints was not noted as it was a pre-testing task.

The system which was used for development and testing of this tool was having following specification.

System architecture:

- RAM size : 2 GB
- Processor : Intel® Core™ 2 Duo CPU @ 2GHz
- Operating System : 32-bit Linux OS (Ubuntu 11.10)
- Compiler: gcc compiler

It is mentioned in former section that the developed algorithms are implemented in 'C' language and the tool was developed using JAVA language. This is because of the following two reasons:

a) Computational speed of 'C' language is faster as compared to that of JAVA

b) To decrease the computation time to much lower value, a parallel approach can be made in 'C' easily using parallel languages like Message Passing Interface (MPI)[22], but this is too difficult in JAVA programming.

As, this tool can also be used for palm print verification or recognition, the larger area size of the palm print image will make the tool slower, that's why a time efficient tool must be developed for the same.

Table-2 shows that the average computation time in case of ART-2 net is more than that of ART-1. This is because the complexity of feed-back path between processing layer of neural nodes and the input layer. A detailed analysis on TSR and EER could be crucial to compare the performance of the two algorithms. Table-3 shows this comparison. For calculation of TSR, the total number of attempts is taken as 740 (total number of patterns in the fingerprint database). However, the time will significantly increase if the supplied fingerprint area is larger.

Table 3 shows that ART-1 is performing with a TSR of **95.80%** and ERR of **15.54%** only when the vigilance parameter is tuned to ***0.93***. The developed ART-1 algorithm outputs the result within 470 milliseconds. Similarly, ART-2 has a TSR of ***97.37%*** and ERR of ***9.72***% at same vigilance parameter. Various plots are given in Fig. 5-9 below to focus more on the comparison of performances between the two algorithms.

In the Fig. 5-7, it can be seen that the tuning vigilance parameter has a major effect on the algorithmic performance. This also accelerates capacity of the tool for detection. With low vigilance parameter, the algorithms are less able to cluster the input patterns and hence are failed to catch skilled imposter pattern. With vigilance parameters tuned between 0.4-0.9, the inputs could be clustered among the set number of cluster units (m=20). Therefore, vigilance parameter near to 0.9 is the optimized value.

Table 1. Showing result of ART-1 algorithm with different vigilance parameter and corresponding FAR, FRR and Average computation time

Tested fingerprint Class	Total Number of attempts (N)	ρ	Total number of Rejected (R-*for Genuine*)/Accepted (A-*for Imposter*)	FRR = $\frac{N(R)}{N}\times 100\%$	Standard Deviation of FRR (±SD)	FAR = $\frac{N(A)}{N}\times 100\%$	Standard Deviation of FAR (±SD)	Avg. Computation Time (ms.)
Genuine	222	0.11	114	51.35	12.153	-	-	340
		0.23	114	51.35				431
		0.33	121	54.50				390
		0.45	103	46.40				461
		0.52	107	48.20				320
		0.61	98	44.14				356
		0.73	83	37.39				390
		0.85	57	25.68				418
		0.89	55	24.77				322
		0.90	58	26.13				380
		0.93	54	24.32				330
		0.99	59	26.58				400
Forged	518	0.11	91	-	-	17.57	3.8523	300
		0.23	84			16.22		370
		0.33	70			13.51		335
		0.45	66			12.74		360
		0.52	51			9.85		458
		0.61	42			8.11		509
		0.73	40			7.72		290
		0.85	49			9.46		364
		0.89	42			8.11		466
		0.90	33			6.37		589
		0.93	35			6.76		610
		0.99	34			6.56		427

Table 2. Showing result of ART-2 algorithm with different vigilance parameter and corresponding FAR, FRR and Average computation time

Tested fingerprint Class	Total Number of attempts (N)	ρ	Total number of Rejected (R-*for Genuine*)/Accepted (A-*for Imposter*)	FRR = $\frac{N(R)}{N}\times 100\%$	Standard Deviation of FRR (±SD)	FAR = $\frac{N(A)}{N}\times 100\%$	Standard Deviation of FAR (±SD)	Avg. Computation Time (ms.)
Genuine	222	0.11	109	49.10	13.455	-	-	510
		0.23	110	49.55				611
		0.33	97	43.69				790
		0.45	99	44.59				543
		0.52	82	36.94				683
		0.61	80	36.04				711
		0.73	83	37.39				737
		0.85	48	21.62				760
		0.89	47	21.17				794
		0.90	39	17.57				822
		0.93	29	13.06				849
		0.99	37	16.67				877
Forged	518	0.11	100	-	-	19.31	5.67	480
		0.23	103			19.88		470
		0.33	105			20.27		535
		0.45	93			17.95		560
		0.52	86			16.60		583
		0.61	68			13.13		618
		0.73	73			14.09		628
		0.85	51			9.85		679
		0.89	39			7.53		715
		0.90	29			5.60		740
		0.93	33			6.37		775
		0.99	36			6.95		821

Table 3. Table showing TSR and ERR and Avg. Computation Time of both ART-1 and ART-2 algorithms

ρ Setting	TSR (%)		ERR (%)		Average Computation Time (ms.)	
	ART-1	ART-2	ART-1	ART-2	ART-1	ART-2
0.11	90.69	90.76	34.46	34.21	320	495
0.23	90.87	90.62	33.79	34.72	400.5	540.5
0.33	90.81	91.36	34.01	31.98	362.5	662.5
0.45	92.01	91.55	29.57	31.27	410.5	551.5
0.52	92.16	92.76	29.03	26.77	389	633
0.61	92.94	93.36	26.13	24.59	432.5	664.5
0.73	93.90	93.04	22.56	25.74	340	682.5
0.85	95.25	95.75	17.57	15.74	391	719.5
0.89	95.56	96.12	16.44	14.35	394	754.5
0.90	95.61	96.87	16.25	11.59	484.5	781
0.93	*95.80*	*97.37*	*15.54*	*9.72*	*470*	*812*
0.99	95.52	96.81	16.57	11.81	413.5	849

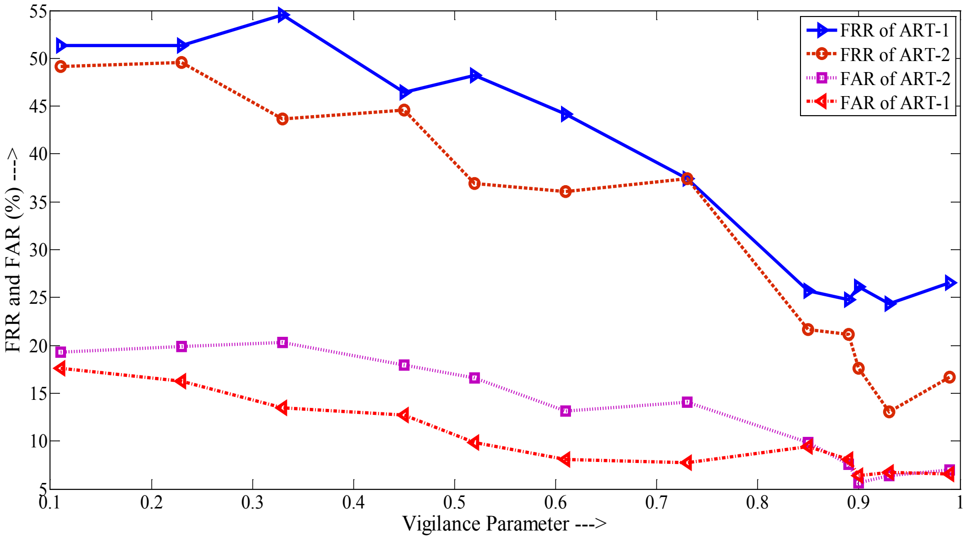


Figure 5. Vigilance parameter vs. FAR and FRR of ART algorithms

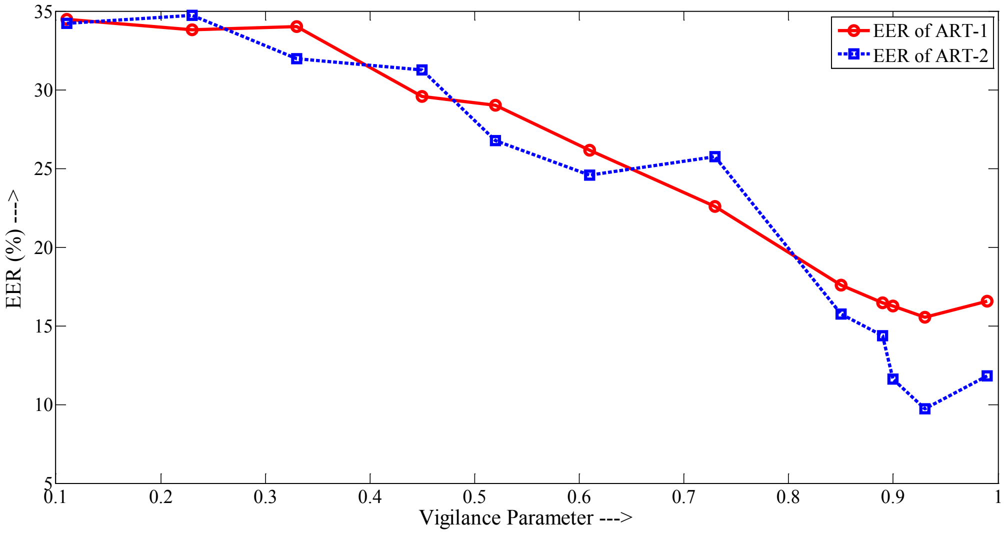


Figure 6. Vigilance Parameter vs. Equal Error Rate

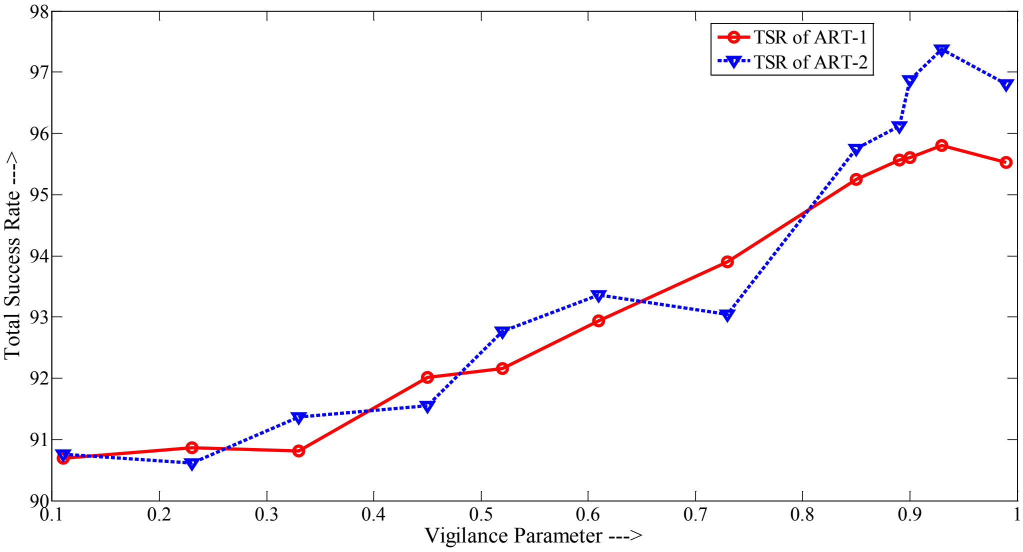


Figure 7. Improvement of Success Rate with vigilance Parameter

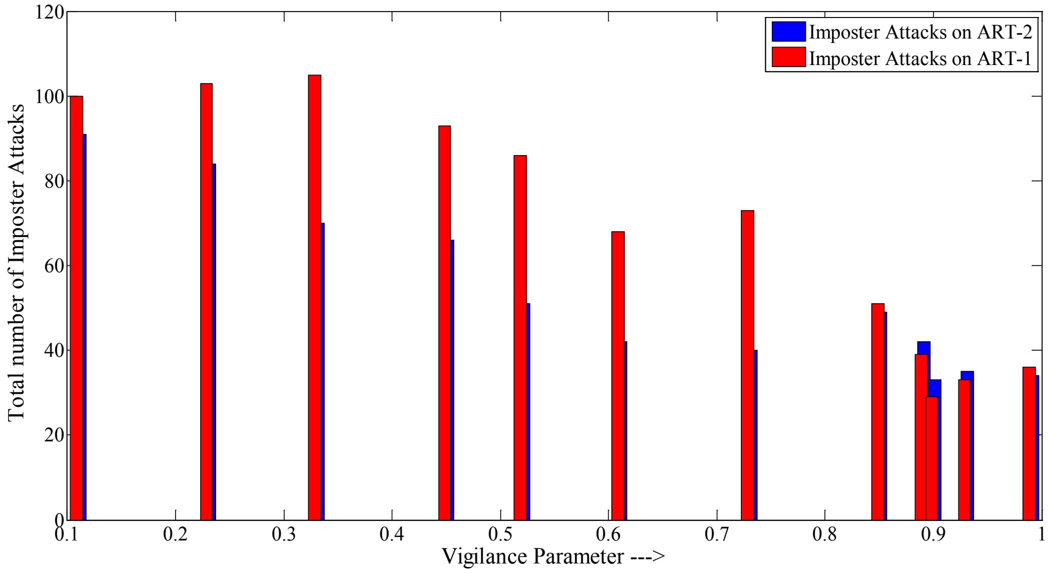


Figure 8. Imposter attacks on ART algorithms and resolution with different vigilance parameters

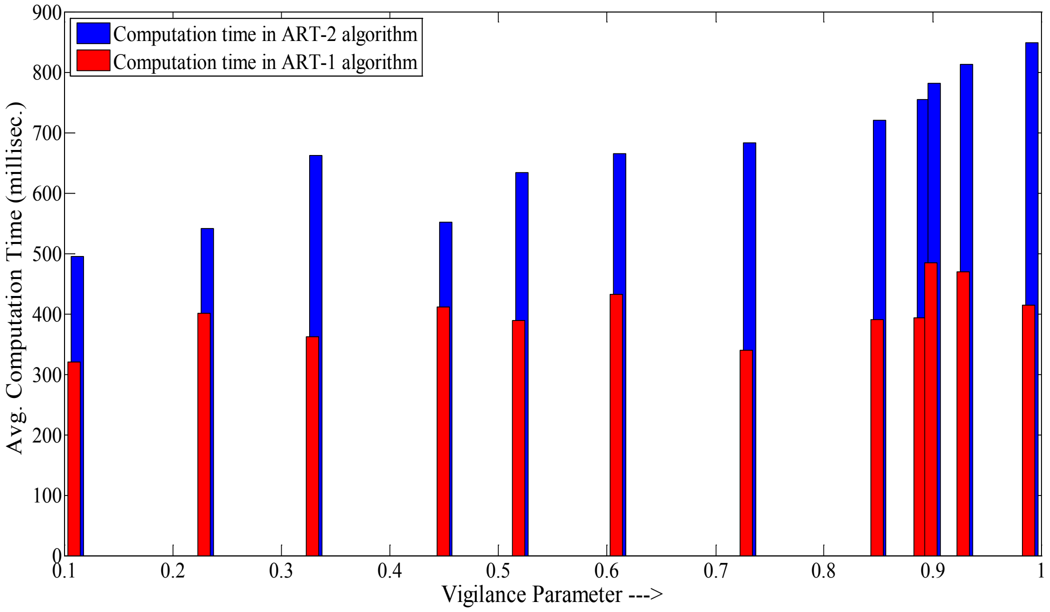


Figure 9. Decision making time taken by two algorithms

Imposter attack is a crucial activity which should be minimized in such biometrics verification systems. In this work (see Fig. 8), ART-1 algorithm with low vigilance is attacked more by imposter patterns than ART-2. However, ART-1 when tested with higher vigilance was seen to be less prone to attacks.

Fig. 9 reveals complexity of ART-2 net has got more effect on decision speed whereas ART-1 is simpler. With increasing vigilance value, ART-2 is doing more computation due to clustering of the patterns to different clusters in the middle layer[3] and hence decreasing the computational speed. However, the computation time may vary from one system to another. So, if we can suppress the delay concept in such comparison, we may claim that ART-2 may perform better than its counterpart, provided the best ‘ρ’ is set.

5. Conclusions and Future Work

In this paper, two algorithms based on ART were proposed for fingerprint verification process. A complete fingerprint tool has been demonstrated with a GUI for testing of the developed algorithm. We have used ART networks – type 1 and 2 to develop the back end of the tool using ‘C’. The front end is developed in JAVA, which has been connected to back end by the concepts of native programming, so that user inputs could be processed and the decision could be viewed. The tool has been tested with 740 test cases of genuine (222) and forged (518) fingerprints. ART-1 achieved a TSR of 95.80% with EER of 15.54% whereas ART-2 performed best with TSR of 97.37% and EER of 9.72% with mentioned vigilance parameter.

From the technical perspective, the paper reveals that tuning of vigilance parameter (ρ) is the key to the successful detection of imposter fingerprints, which has been performed during the training. In this study, ART-1 and ART-2 with ρ = 0.93 detect very similar looking forged patterns with maximum success rates. The study also observes that, in case of '*n*' number of pixels partitioned into '*m*' number of clusters (in case of ART-2), the algorithmic time complexity becomes O(n × m). This issue could be handled with the help of parallel computing, so that the said tool could be more time efficient.

The authors are currently working on various standardization processes of the tool for its future implementations.

REFERENCES

[1] BHANU, B., TAN, X., AND MEMBER, S. (2003). Fingerprint indexing based on novel features of minutiae triplets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25, 616–622.

[2] PENG, L., YANG, X., QIAO, H., CAO, K., LIU, E., AND TIAN, J. (2012). An effective biometric cryptosystem combining fingerprints with error correction codes. *Expert Systems with Applications*, 39, 6562-6574.

[3] SAHU S.K. SAHANI S.. JENA P.K.. CHATTOPADHYAY S. (2012). Fingerprint Identification System using Tree based Matching. *International Journal of Computer Applications*, 53, 11-16.

[4] SIVANANDAM, S.N., AND DEEPA, S.N. (2011). Principle of Soft Computing. 2nd ed. Wiley-India. ISBN-978-81-265-2741-0.

[5] CHATTOPADHYAY S., KAUR P., RABHI F.A., AND ACHARYA U.R. (2011b). An Automated System to Diagnose the Severity of Adult Depression". In the proceedings of *2nd International Conference on Emerging Applications of Information Technology* (CSI EAIT-2011), 19-20th February Kolkata India, (Editors - Jana D., and Pal P.), pp. 121-124. Publishers: IEEE Computer Society and Conference Publishing Services.

[6] CHATTOPADHYAY S. (2012). Neurofuzzy model to automate the grading of old-age depression, *Expert Systems: the Journal of Knowledge Engineering*, accepted (in press)

[7] CHATTOPADHYAY S., KAUR P., RABHI F., AND ACHARYA U. R. (2011a). Neural Network Approaches to Grade Adult Depression, *Journal of Medical Systems* (in press), DOI: 10.1007/s10916-011-9759-1.

[8] DASH, T., NAYAK, T., AND CHATTOPADHYAY, S. (2012a). Offline Verification of Hand Written Signature Using Adaptive Resonance Theory Net (Type-1). In the proceedings of *4th Int. Conf. of Electronics Computer Technology*, Vol-2, (Editor: Yuan Li): 205-210.

[9] DASH, T., NAYAK, T., AND CHATTOPADHYAY, S. (2012b). Offline Verification of Hand Written Signature Using Auto Associative Memory Net. *International Journal of Advanced Research in Computer Engineering & Technology*, **1**(4), 370-374.

[10] YANG, J.C., AND PARK, D.S. (2008). A fingerprint verification algorithm based on tessellated invariant moment features. *Neurocomputing*, 71, 1939-1946.

[11] DASH T.. NAYAK T.. CHATTOPADHYAY S. (2012). Handwritten Signature Verification (Offline) using Neural Network Approaches: A Comparative Study. *International Journal of Computer Applications*, 57, 33-41.

[12] LI, P., YANG, X., QIAO, H., CAO, K., LIE, E., AND TIAN, J. (2012). An effective biometric cryptosystem combining fingerprints with error correction codes. *Expert Systems with Applications*, 39, 6562–6574.

[13] MALI, K., AND BHATTACHARYA, S. (2011). Fingerprint Recognition system based on Global and local structures. *International Journal of Computer Science and Engineering*, **3**(1), 161-172.

[14] ROSS, A., AND JAIN, A. (2003). Information Fusion in Biometrics. *Pattern Recognition Letters*. 24 (13), 2115-2125.

[15] PRABHAKAR, S., AND JAIN, A.K. (2002). Decision-level fusion in fingerprint verification. *Pattern Recognition*, 35, 861-874.

[16] CAPPELLI, R., AND FERRARA, M. (2012). A fingerprint retrieval system based on level-1 and level-2 features. *Expert Systems with Applications*, 39, 10465–10478.

[17] JIN, A.T.B., LING, D.N.C., AND SONG, O.T. (2004). An efficient fingerprint verification system using integrated wavelet and Fourier–Mellin invariant transform. *Image and Vision Computing*, 22, 503–513.

[18] MAIO, D., MALTONI, D., CAPPELLI, R., WAYMAN, J.L., AND JAIN, A.K. (2002). FVC2002: Fingerprint Verification Competition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3), 402-412.

[19] BAZEN, A.M., VERWAAIJEN, G.T.B., GEREZ S.H., VEELENTURF L.P.J, AND ZWAAG B.J, (2000). A Correlation-Based Fingerprint Verification System. *Proceedings of the ProRISC/IEEE workshop (STW-2000),* 205-213.

[20] CAPPELLI, R., MAIO, D., MALTONI, D., WAYMAN, J.L., AND JAIN, A.K. (2006). Performance Evaluation of Fingerprint Verification Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1), 3-18.

[21] SCHILDT, H., (2012). JAVA: The complete Reference, 7th Edition. Tata McGraw Hill Education Pvt. Ltd. ISBN-10: 0-07-063677-X.

[22] M.J. QUINN (2003). Parallel Programming in C with MPI and OpenMP. International Edition. McGraw-Hill Higher Education, ISBM: 007-282256-2.