Handwritten Signature Verification (Offline) using Neural Network Approaches: A Comparative Study

Tirtharaj Dash
Dept. of Computer Science. & Engineering
Veer Surendra Sai University of Technology,
Burla-768018, India

Tanistha Nayak
Dept. of Information Technology
National Institute of Science and Technology,
Berhampur-761008, India

Subhagata Chattopadhyay
Bankura Unnayani Institute of Engineering,
Bankura-722146, West Bengal, India

ABSTRACT

Forgery detection has been a challenging area in the field of biometry, e.g., handwritten signatures. Signature verification is a bi-objective optimization problem. The two crucial parameters are accuracy and time of computation. In this work, a comprehensive study on application of Adaptive Resonance Theory (ART) Nets (Type 1 and 2) and Associative Memory Net (AMN) has been conducted. To decrease the time complexity a corresponding parallel version using OpenMP is developed for each algorithm. The algorithms are trained with the original/genuine signature and tested with a sample of twelve very similar-looking forged signatures. The study concludes that ART-1 detects fake signatures with an accuracy of 99.89%; whereas, ART-2 and AMN detect forgery with accuracies of 99.99% and 75.68% respectively which are comparable to other methods cited in this paper.

General Terms

Pattern Recognition, Neural Networks, Soft Computing, Parallel Processing

Keywords

Forgery detection; signature verification; bi-objective optimization; Adaptive Resonance Theory; Associative Memory Net

1. INTRODUCTION

Handwritten signature is the most commonly used biometric techniques for the personal verification/identification.[1] Skillful copy of a signature is not uncommon. It leads to various legal issues. Detecting such a false but very similar looking signature is quite challenging in machine intelligence research. To address this issue, various signature detection schemes are being proposed, which include both traditional approaches and soft computing techniques, which have been discussed in section 2.

Neural Network (NN) learns patterns by examples or observation [1]. The said learning can be Supervised or Unsupervised. Adaptive Linear Net (ADALIN), Multiple ADALIN (MADALIN), Perceptron Network, etc. are some important examples of supervised learning method. These NNs learn faster and accurately, but the problem with these is that, new training is required each time it learns 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. 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. Another NN to be known as Associative Memory Net (AMN) updates its knowledge base through the concept of supervised learning. However, it operates on the principle of unsupervised learning method. Depending on target pattern, AMN can be Auto-AMN or Hetero-AMN. In the former type, target is similar to the training pattern. This concept differentiates itself from its counterpart.

In this study, we have developed three algorithms based on principle of ART-1, ART-2 and Auto-AMN. As signature verification is a bi-objective optimization problem where highest accuracy should be achieved with less time consumption, for all the three algorithms parallel methods have been proposed. The parallel implementation distributes the computation work to multiple processors and achieves the same result in a minimal CPU time. The verification is performed offline. We have also compared the performances of ART-1, ART-2 and AMN.

2. LITERATURE REVIEW

As mentioned above, uncountable techniques have been proposed by researchers in the field of signature verification. It is not feasible to describe all. Still, we have researched on various traditional and soft computing-based methods. The major sources of the literatures are Google Scholar, Scopus, Science Direct, and IEEE Xplore. Relevant studies are briefly discussed below.

A *Bayesian network* representation has been proposed by Xiao and Leedham in 2002 to handle the uncertainty related to the match of forged and original signatures. The authors have proposed a decision tree like network, where each node of the tree computes the conditional probability as the chance of matching [2].

Another method, known as *displacement extraction method* has been proposed for offline signature verification [3]. In this work, the authors have extracted a displacement function between a pair of signatures, original and fake. The study concludes that the average accuracy rate of detection is around 75%.

In the year 2005, Kholmatov and Yanikoglu presented an online signature recognition scheme with overall accuracy rate of 98.6%. The authors have used *three dimensional feature vector* and *Dynamic Time Warping (DTW)* [4].

It is observed that the method outperforms when both static and dynamic features of the signatures are taken into consideration for verification. Based on this fact, an *optimal* *function* of features was used for online verification [5]. In this work, the authors first choose a candidate function and optimized it to produce an optimal function. Basically the optimization was done using well known *Genetic Algorithm (GA)*. The error rate in this work was only 0.1%.

Parametric and Reference pattern based features (RPBF) was used to simulate handwritten signature verification system. It was found that the RPBF improves the results to 90% if only shape feature of the signature was taken. It was also deduced that the result can be improved to more extent if two dimensional RPBF is taken into test [6].

Cooperating NN was used for offline signature verification. The features used in this work were geometrical parameters, outline and image of the signature. The accuracy rate in this work was 96% [7].

Wavelet thinning features were used for offline signature verification using *Matching Algorithm*. Global and local alignment algorithms were used to define structure distortion using signature skeleton. Similarity measurement was evaluated using Euclidean distance of all found corresponding feature points. The accuracy in this case was 81.4% [8].

A different method was proposed using smoothness of curve. The method suggested that the *cursive segment* of forged signature is less smooth than that of genuine one. Two methods were proposed to extract the smoothness feature: a crossing method and fractal dimension method. Satisfactory results were also obtained when this technique was combined with global shape feature [9].

Hidden Markov Model (HMM) was successfully used for signature verification. It was performed by analysis of alphabets within the signature. According to this model, a signature is collection of vectors related each point in its outline. The average accuracy rate was 88.9% in this case [10].

A novel approach for offline signature verification and identification was proposed using *distance statistics*. It used quasi-multiresolution technique using Gradient, structural and concavity features. The method yielded an accuracy of 78% for verification and 93% for signature identification [11].

Singular Value Decomposition using data glove was used successfully for signature verification by Kamel and Sayeed (2008). [12] In this work, SVD was used in finding 'r' singular vectors sensing the maximal energy of glove data matrix, called principal subspace. After identifying data glove signature through its rth principal subspace, the authenticity can be obtained by calculating the angles between the different subspaces [12].

A method based *histogram processing* was used for offline signature verification [13]. The method first extracts robust Edge Orientation Distance Histogram descriptor which reflects the signature structure variation. In addition to this, directional gradient density features were employed for skilled forgery verification. The method achieved improved accuracy.

Signature verification using *"Siamese" Time-Delay NN* was proposed. However, the algorithm was based on ANN. Siamese time delay NN stands for two identical NNs joined at their outputs. The complete algorithm was based on training and testing of the NNs. First the genuine signature was training the networks and the forgeries were being used for testing the NNs for matching [14].

A new approach to Japanese signature verification was proposed which eliminated *background pattern*. In this method to convert the signature to binary form subtraction and thresholding method were used. The found error rate was 14% [15].

Statistical method was adopted for online signature verification. Shape and dynamic features were used as feature and Euclidean distance method was used for error calculation. The performance of Euclidean distance was superior on large set of data. The method accepted 99.5% of genuine signatures and rejected 90% of forged signature making the average accuracy to 90% [16].

A combination of *shape contexts* and local features were used for online signature verification. However, *DTW* technique was also used for elastic matching between signatures. The proposed method suggested that by combining local features with shape contexts the performance of the algorithm could be increased. The average error rate in this work was 6.77% [17].

A new method signature verification system based on *rule based inductive learning system (3-ext inductive learning system)* proposed by Aksoy and Mathkour.[18] It was a hybrid technique and used template matching for feature extraction and 3-ext inductive learning algorithm to extract the rules and verification of the signatures. The ability to correctly classify the signatures based on these techniques was 97% [18].

Another method for signature verification using *NN* was developed. The work presented an analysis of Hu's moment invariants on image scaling and rotation. From this work the authors found that image's spatial resolution is important to store the invariant features. In addition to it, it was also found that decrease the fluctuation of moment invariants, the image spatial resolution should be higher than the level of image scaling and rotation. Experimental results showed that the computation time increases with resolution of the image used for the verification [19].

For signature verification, methods based on ART are also being proposed. The method was based on 1-bit quantized pressure pattern in time domain. The work suggested that this timing information can be used for first stage screening of incoming signatures using ART-1 networks with various values of vigilance parameter [20].

A combination of ART-2 and Fast Wavelet Transform (FWT) was used for signature verification. In this work, FWT was employed for feature extraction. The authentic data was used for training of ART-2 net and forged data was used for verification purpose [21].

Dash et al., (2012) studied ART-2 Net [22] and Associative Memory Net [23] for recognizing the very similar-looking but forged signatures. The authors found that ART-2 Net is capable of recognizing such signatures with almost 100% accuracy [22]; while with the second technique the accuracy was 92.3% [23].

3. METHODOLOGY

A step-wise method has been followed in this work. The steps include:

Step-1: Collection of the signature samples (Original as well as Forgeries)

Step-2: Feature Extraction

Step-3: Implementation of ART-1, ART-2 and AMN networks in '*C*' language

Step-4: Training of the networks with genuine signatures

Step-5: Testing the networks with twelve very similar looking forged signatures, and

Step-6: Comparison of the performance the networks based on

(i) Detection accuracy

(ii) CPU-time in

a) Serial Implementation

b) Parallel implementation

3.1 Collection of Signature Samples

Original signature is produced at first and then forged signature samples are collected from twelve different persons at different times (refer to Appendix-I). Each person has been given ample time (1 month each) to enable copying the original signature with almost no visually detectable mistakes. The signature templates are shown in Appendix-I. All signatures are then saved into BMP files of size: 200×63dpi with Bit depth as 4.

3.2 Feature Extraction

A user-defined image function in *C-Program* is used for extraction of pixels from the BMP files. The method of feature extraction is given in Fig-1.

A sample structure of the BITMAP header is given in a piece of code below.

```
typedef struct {
   unsigned int width;
   unsigned int height;
   unsigned int planes;
   unsigned short bitcount;
   unsigned int size;
} BITMAPINFOHEADER;
```

A pixel can be characterized by three colors: Red, Green and Blue (RGB). The corresponding structure is given below.

```
typedef struct {
   unsigned char blue;
   unsigned char green;
   unsigned char red;
} PIXEL;
```

Sample pixel values of the original and one forged signatures are given in Appendix-II.

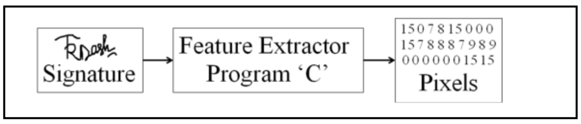

Fig-1. Feature Extraction using our C-program.

3.3 Implementation of ART & AMN Nets and their Trainings

3.3.1 A general algorithm for ART net implementation and its training

```
START
Initialize learning rate (α), vigilance
parameter (ρ), initial weights (b_ij(0),
t_ji(0))
WHILE (Stopping condition is FALSE)
DO
        FOR each input vector
        DO
        Extract pixels from original sign.

        F1-Layer Processing
        IF (Reset is TRUE)
               Find the victim unit (F2
        unit) to learn the current input
        Pattern

               Calculate F1(b) unit from
        F1(a) and F2
        ELSE
               Perform weight updation
        END
END
REPEAT for the tested forged signature.
STOP
```

Updation of weights is performed using the following mathematical relations:

Equations 1, 2 correspond to the bottom-up and top-down weight updation respectively in ART-1.

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

$$t_{ji}(new) = x_i \qquad (2)$$

Similarly, Equations 3, 4 refers the weights of ART-2 net.

$$b_{iJ}(new) = \alpha d u_i + \{[1 + \alpha d(d-1)]\} b_{iJ} \qquad (3)$$

$$t_{iJ}(new) = \alpha d u_i + \{[1 + \alpha d(d-1)]\} t_{iJ} \qquad (4)$$

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

$t_{iJ}(new)$ = 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 5 (ART-1), 6 (ART-2).

$$\|x\| = \sum_{i=1}^{n} x_i \qquad (5)$$

$$\|x\| = \sqrt{\sum_{i=1}^{n} x_i^2} \qquad (6)$$

3.3.2 AMN implementation algorithm and its training

```
INITIALIZE weight (W) to 0

INPUT the original sign. to the first
layer of AMN

FOR i=1 to n

DO

        FOR j=1 to n

        DO

        CALCULATE the weight as

        Wij(new)=Wij (old) + INPUTi×TARGETj

        END

END

FOR i=1 to n

DO

        FOR j=1 to n

        DO

        CALCULATE the net input to each
        output node as,
```

$$Yin_j = \sum_{i=1}^{n} x_i W_{ij}$$

```
        IF(Yinj>0)

                Yj=+1;

        ELSE

                Yj=-1;

        END

END
```

The training method is represented in Fig-2.

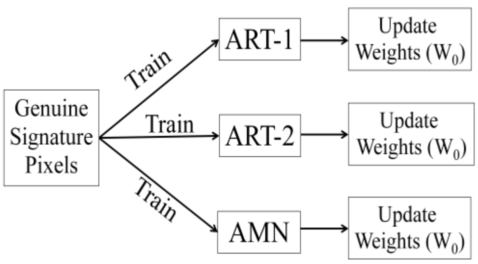

Fig-2. Training the networks with Original Signatures

3.3.3 Parallel Implementation of the Networks

It is better to mention that the method should give maximum accuracy by utilizing the CPU for minimum time. So, in this work, we also developed parallel version of all the above algorithms using OpenMP (www.openmp.org). By doing this, three major things could be achieved; (i) reduction in computation time, (ii) utilization of all the processor of the system and (iii) inherent parallelism property of NN could be used.

However, the parallelization can be done where there is loop over independent instructions. To have parallelization in the loop, OpenMP directives are used [24].

The parallel programming directive used is Row Block algorithm which can be expressed as follows.

#pragma omp parallel

for(…)

{

…

…

}

The function used to calculate the time details in Sequential implementation is **clock()** and in parallel implementation is **omp_get_wtime()**. The header file used in the parallel programming is **omp.h**.

3.3.4 System architecture specification

The implementation (both training and testing) are carried out in a system having **Intel Quad Core Processor** with **2GB RAM** having a processor speed of **2GHz**. The operating system used is **Linux (Ubuntu Version 10.10)**. It is mentioned that the package used for parallel processing is the *OpenMP 3.0*.

3.4 Testing trained nets with Forged Samples

The trained networks are then tested with forged signatures. The testing method is represented in the Fig-3.

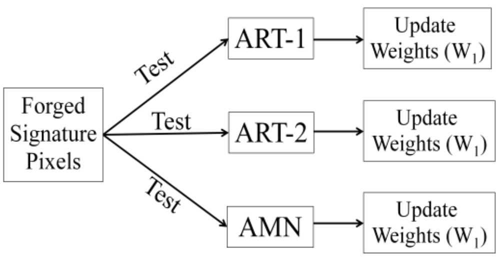

Fig-3. Testing of the trained networks

3.4.1 Calculation mismatch in ART nets

After passing each of the forged signatures through the trained ART networks and the number of matched b_{ij} is counted. Equation-7 expresses the mismatch percentage as follows,

$$mismatch = \left[1 - \frac{b^{*}_{ij}}{count}\right] \times 100 \qquad (7)$$

In this equation, '*count*' denotes the total number of bottom-up weights (b_{ij}) and 'b^*_{ij}' are the weights which are matched with the training cases.

3.4.2 Calculation of mismatch in AMN

The numbers of Y_J which are -1 are counted (count) (*see Section 3.3.2*). The mismatch percentage can be calculated using the following Equation-8.

$$Mismatch = [count/Total\ Number\ of\ Pixels] \times 100 \qquad (8)$$

3.4.3 Setting the threshold of mismatch

Threshold is the minimum mismatch percentage after which the tested signature could be termed as illegal. As the key task behind this work is to impose stricter security and safety applications, we set the threshold as low as **5%** to avoid acceptance of highly skilled forgeries. That is, in case any mismatch of more than or equal to 5%, the tested signature will be rejected. It is important to note that, such threshold setting must be situation specific and the choice of the administrator/user.

3.5 Comparison between Results of ART-1, ART-2 and AMN Networks

The results of all the three techniques are compared with respect to (i) accuracy (ii) CPU time as mentioned earlier in this section. It is important to note that, in the ART implementation vigilance parameter (ρ) and number of cluster units (m) is varied to obtain best mismatch in the result. Experimental results are showed in section 4. It is also important to mention that the time of computation in ART-1, ART-2 and AMN techniques are compared for (i) sequential as well as (ii) parallel processing. As the work involves real world data, the challenge is to obtain results with minimal error and less computation time.

4. RESULTS AND DISCUSSION

The contribution of our work can be divided into two major parts; (i) method of signature verification using ART nets (ii) method of signature verification using AMN. It is wise to note that all the implementation is carried out using both serial and parallel processing techniques.

- In our ART (ART-1, 2) techniques, a detailed parametric study on 'ρ' i.e. vigilance parameter with sequential and parallel processing has been provided.
- In the AMN technique, study on signature area and its affect on computation time is provided.

4.1 Performance of ART-1 Techniques

The following sections give a detailed study on signature verification method using ART-1 technique.

4.1.1 Accuracy check with various vigilance parameter (ρ)

Table-1 shows mismatch (%) between the original and each of the forged signatures with ART-1 technique with various values of 'ρ' ranging from 0.50 to 0.99 as 0<ρ<1. It may be noted that for decision making, we have set the threshold as 5% mismatch. Hence, any mismatch threshold <5% is considered as 'accept' and vice versa. That is why; the forged signatures 11 and 12 are accepted. By setting a stricter mismatch threshold, such as <1%, these false acceptances could be avoided. This is also the case of ART-2 technique as both the techniques are homomorpic to each other. It should be noted that, in both ART-1 and 2 techniques the number of cluster units 'm' is set to 20.

Table 1. Mismatch Percentage at different vigilance parameter in ART-1 Technique

Test Case (Original vs.)	**Mismatch (%) at Vigilance Parameter (*ρ*)**						
	0.50	***0.63***	***0.78***	***0.89***	***0.97***	***0.99***	**Decision**
Original	0.061	0.077	0.065	0.002	0.090	0.092	*Accepted*
Forged 1	20.692	20.676	20.658	20.751	20.663	20.665	Rejected
Forged 2	19.930	19.914	19.896	**19.991**	19.901	19.906	Rejected
Forged 3	22.303	22.297	22.269	**22.365**	22.281	22.283	Rejected
Forged 4	19.184	19.177	19.150	**19.246**	19.199	19.189	Rejected
Forged 5	17.541	17.525	17.517	17.601	17.497	17.465	Rejected
Forged 6	17.168	17.152	17.134	**17.228**	17.145	17.147	Rejected
Forged 7	20.811	20.775	20.756	20.870	20.786	20.781	Rejected
Forged 8	20.636	20.620	20.756	**20.690**	20.612	20.781	Rejected
Forged 9	23.414	23.407	23.387	23.474	23.417	23.416	Rejected
Forged 10	21.057	21.041	21.113	**21.117**	21.028	21.023	Rejected
Forged 11	*1.073*	*1.057*	*1.036*	*1.135*	*1.044*	*1.067*	*Accepted*
Forged 12	*2.303*	*2.287*	*2.289*	*2.363*	*2.274*	*2.271*	*Accepted*

[Bold values matches with the ideal results]

From Table-1 it is clear that the ART-1 net performs to its best when vigilance parameter 'ρ' is tuned to 0.89. The net gives the exact result for Forged signatures 2, 3, 4, 6, 8 and 10. The results match with the ideal results which are calculated using similarity index (SI) between original and the considered forged signature. SI can be calculated using Equation-9.

$$SI = \frac{1 - D_p}{T_p} \times 100 \qquad (9)$$

Where, D_p is the number of 'dissimilar pixels' and T_p is the total number of pixels in the signature image.

4.1.2 Computation time consumption with various ρ:

The computation time is the sum total of time taken by *feature extraction, training of the net and testing* each forged signatures. The parallel program is run in *Quad core* system (see System specification, section 3.3.4). Table-2 shows accuracy and timing analysis with different values of vigilance parameter. It will be wise to note that the accuracy is calculated with respect to the ideal mismatch result.

Table 2. Accuracy and Computation time analysis with different 'ρ' for ART-1 method

ρ	Accuracy (%)	Computation Time (seconds)	
		Sequential	Parallel
0.50	99.67	4.87	1.49
0.63	99.59	5.23	1.55
0.78	99.60	7.03	2.26
0.89	***99.89***	*8.37*	*2.28*
0.97	99.55	7.12	2.08
0.99	99.54	9.04	2.72

It can be seen that setting the vigilance parameter to 0.89, an average accuracy of *99.89%* is achieved. The computation time at this vigilance parameter is 8.37 sec in serial implementation and 2.28 sec in parallel implementation. If the application needs a slightly lower accuracy then vigilance parameter can be tuned to 50% and a lowest computation time of 1.49 sec in parallel can be achieved.

4.2 Performance of ART-2 Technique

4.2.1 Result check with various vigilance parameter

Table-3 shows the results of ART-2 technique with different values of vigilance parameter.

Table 3. Mismatch Percentage at different vigilance parameter in ART-2 Technique

Test Cases (Original vs.)	Mismatch (%) at Vigilance Parameter (*ρ*)						
	0.50	***0.63***	***0.78***	***0.89***	***0.97***	***0.99***	**Decision**
Original	0.004	0.005	0.006	0.006	0.003	0.008	*Accepted*
Forged1	20.758	20.759	20.760	20.761	20.754	20.762	Rejected
Forged2	19.996	19.997	19.998	19.999	**19.991**	20.000	Rejected
Forged3	22.369	22.370	22.371	22.372	**22.365**	22.373	Rejected
Forged4	19.250	19.251	19.252	19.253	**19.246**	19.254	Rejected
Forged5	17.607	17.608	17.609	17.600	17.603	17.611	Rejected
Forged6	17.233	12.235	17.236	17.237	17.230	17.238	Rejected
Forged7	20.867	20.878	20.879	20.870	20.873	20.881	Rejected
Forged8	20.702	20.703	20.705	20.705	**20.690**	20.706	Rejected
Forged9	23.480	23.481	23.482	23.483	**23.480**	23.484	Rejected
Forged10	21.123	21.124	21.125	21.126	21.120	21.127	Rejected
Forged11	*1.139*	*1.138*	*1.140*	*1.132*	***1.135***	*1.136*	*Accepted*
Forged12	*2.369*	*2.370*	*2.371*	*2.262*	*2.365*	*2.368*	*Accepted*

[Bold values matches with the ideal results.]

Observing the Table-3, the ART-1 net gives exact mismatch at ρ=0.97. A total of six results are matching with the ideal mismatch. The signatures which got verified with 100% accuracy are Forged signatures 2, 3, 4, 8, 9, 11.

4.2.2 Computation time consumption with various vigilance parameter

Table-4 shows accuracy and timing analysis with different values of vigilance parameter.

Table 4. Accuracy and Computation time analysis with different 'ρ' for ART-2 method

ρ	Accuracy (%)	Computation Time (seconds)	
		Sequential	Parallel
0.50	96.01	5.96	1.78
0.63	97.09	6.72	2.10
0.78	97.61	7.32	1.98
0.89	98.97	4.39	1.41
0.97	***99.99***	*5.86*	*1.58*
0.99	99.91	8.03	2.31

Comparing the result of ART-2 with accuracy of ART-1, it can be seen that the ART-2 net performs with *99.99%* accuracy at ρ=0.97. The time consumed in this setting is 5.86 sec in serial and 1.58 sec in parallel implementation. But observing the vigilance parameter range from 0.50-0.89, ART-1 net performs at 99% or more where as ART-2 detects the forgery with ranging from 96.01%-98.97% respectively.

4.3 Comparison of ART-1 and ART-2 Techniques

4.3.1 Comparison with respect to accuracy

The ART-1, 2 methods are compared with respect to accuracy with different values of vigilance parameter. A plot has been given Fig-4 to view the difference.

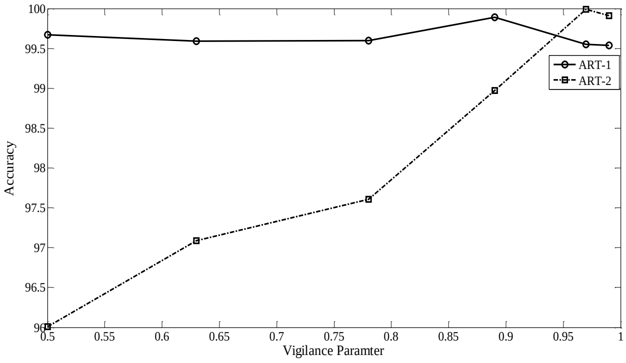

Fig 4. Plot showing accuracy of ART-1 and ART-2 Techniques

4.3.2 Comparison with respect to Computation time

The ART-1, 2 methods are compared with respect to processing with different values of vigilance parameter. A plot has been given Fig-5 to view the difference.

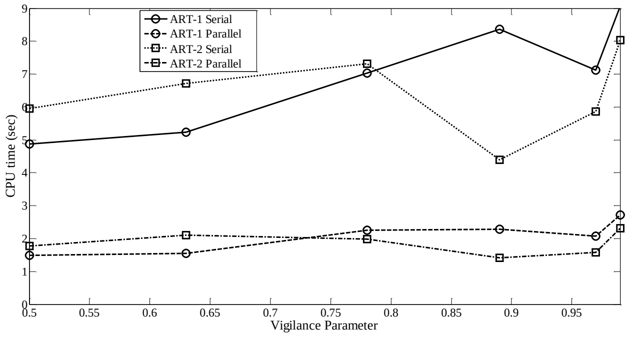

Fig 5. Time consumption by ART-1 and ART-2 method in Serial and Parallel implementations

Clearly studying Fig-4, we can claim that the parallel implementation speeds up the computation and gives the result with in 2 sec only. If number of processor in the system increases the computation time decreases proportionately.

4.4 Performance of AMN Technique

It should be noted that the AMN network implemented here is auto-associative. The reason is that the forged signature will be checked with the genuine version of itself. So the network becomes more complex due to increase in pixels of the signature image. AMN is a two layer NN with the first layer being the pixels values of the forged signatures and second layer contains the pixel nodes from the original signature.

4.4.1 Mismatch results of AMN Network

A result data has been given in Table-5 to view the mismatch outcome and corresponding computation time by the AMN algorithm.

Table 5. Mismatch obtained from AMN network testing

Test Case (Original vs.)	Mismatch (%)	Decision	Computation Time (seconds)	
			Serial	Parallel
Original	21.312	*Reject*	6.15	1.98
Forged1	29.288	Reject	7.9	2.44
Forged2	28.922	Reject	8.8	2.67
Forged3	29.859	Reject	12.06	3.38
Forged4	28.204	Reject	7.13	1.80
Forged5	28.437	Reject	10.24	2.87
Forged6	28.662	Reject	10.85	2.10
Forged7	29.647	Reject	11.46	3.02
Forged8	29.077	Reject	12.08	4.33
Forged9	31.020	Reject	12.69	4.54
Forged10	29.091	Reject	6.304	2.00
Forged11	21.572	*Reject*	7.91	2.89
Forged12	22.093	*Reject*	14.52	4.83

Looking at the mismatch results of AMN from Table-5, it can be said that the network is performing better for all the forged signatures (Forged 1-12), which was not the case of ART techniques. In this statement, AMN may be claimed to the superior of ART networks. But the network is rejecting the original signature itself. The average time taken by the serial algorithm is 9.85 seconds where as the parallel algorithm takes only 2.98 seconds.

4.4.2 Acceptance accuracy of each forged signature

A plot has been given in Fig-6 to see which one of the forged signatures sample is getting verified with highest accuracy.

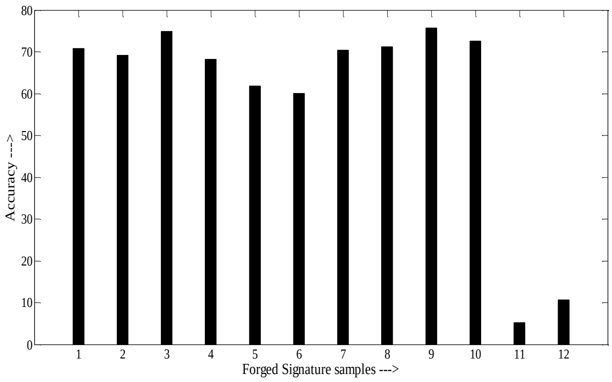

Fig 6. Detection accuracy of AMN for each forged signatures

From the plot above, it is clear that the sample of the type 9 can be detected as forged with highest accuracy of ***75.68%***.

4.4.3 Forged Signature versus Computation time plot

A plot is given in Fig.7 showing CPU utilization time for AMN to recognize each of the forged signature samples.

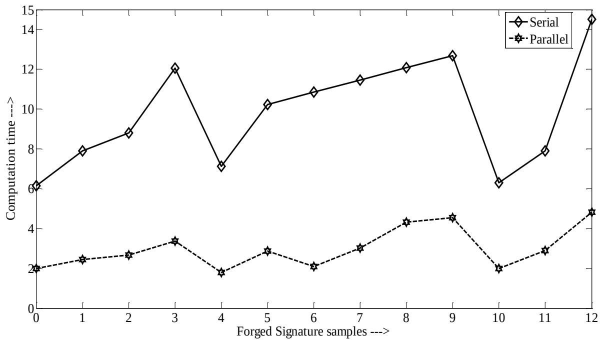

Fig 7. Time taken by AMN to detect each forged signatures

4.5 Comparison of Performance of Difference Methods in Handwritten signature verification

The following table, Table-6 reveals the standing of our methods with respect to accuracy when these are compared to other proposed methods.

Table 6. Comparison of different methods based on accuracy

sl. no	Techniques	Year	Reference	Accuracy
1	Displacement Method	2002	[3]	75%
2	DTW	2005	[4]	98.6%
3	Hybrid of Optimal Function and GA	2004	[5]	99.9%
4	Cooperative NN	1994	[7]	96%
5	Matching Algorithm	2007	[8]	81.4%
6	HMM	1994	[10]	88.9%
7	Distance Statistics	2004	[11]	78%
8	Background Pattern Method	1994	[15]	86%
9	Statistical Method	1994	[16]	90%
10	Hybrid of Shape contexts and DTW	2006	[17]	93.2%
11	3-ext inductive learning system	2011	[18]	97%
12	ART-1 *(our work)*	2012	-	*99.89%*
13	ART-2 *(our work)*	2012	-	*99.99%*
14	AMN *(our work)*	2012	-	*75.68%*

5. CONCLUSION

In this paper, we proposed three adaptive neural network techniques for offline hand-written signature verification. All the three methods have been optimized with respect to accuracy and computation time. We proposed two algorithms for each technique, serial and parallel. The study revealed that ART-1 with ρ=0.89 gives 99.89% accuracy by consuming 8.37 seconds in serial and 2.28 seconds in parallel. For the case of ART-2, ρ=0.97 achieves 99.99% accuracy and time consumed is 5.86 seconds in serial and 1.58 seconds in parallel execution in a quad-core processor. For both ART-1 and ART-2 number of cluster units '*m*' was set to 20. AMN takes an average of 9.58 seconds in serial and 2.98 seconds in parallel to give a detection accuracy of 75.68%. The error threshold is set to 5% in this work to make decision. However, it is completely application based setting.

It is important to mention that in this work we have tested our algorithms on a sample of only twelve forged signatures. It will be impressive to test the algorithms with more training and test cases and then develop a GUI for the applications to make it user friendly.

6. REFERENCES

[1] T. Dash, T. Nayak, S. Chattopadhyay, Offline Verification of Hand Written Signature Using Adaptive Resonance Theory Net (Type-1), in Proc: *IEEE Int. Conf. Electronics Computer Technology (ICECT)* (2012), Vol-2, pp. 205-210.

[2] X. Xiao, G. Leedham, Signature verification using a modified Bayesian network, *Pattern Recognition* (2002) 35, 983-995.

[3] Y. Mizukami, M. Yoshimura, H. Miike, I. Yoshimura, An off-line signature verification system using an extracted displacement function, *Pattern Recognition Letters* (2002), 23, 1569-1577.

[4] A. Kholmatov, B. Yanikoglu, Identity authentication using improved online signature verification method, *Pattern Recognition Letters* (2005), 26, 1400-2408.

[5] J.C.M. Romo, R.A. Silva, Optimal Prototype functions of features for online signature verification, *International Journal of Pattern Recognition and Artificial Intelligence,* Volume: 18, Issue: 7(2004) pp. 1189-1206.

[6] M. Ammar, Progress in verification of skillfully simulated handwritten signatures, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 5, Issues: 1-2(1991) pp. 337-351.

[7] H. Cardot, M. Revenu, B. Victorri, M.J. Revillet, A static signature verification system based on cooperating neural networks architecture, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 8, Issue: 3(1994) pp. 679-692.

[8] B. Fang, X. You, W.S. Chen, Y.Y. Tang, Matching algorithm using wavelet thinning features for offline signature verification, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 5, Issue: 1(2007) pp. 27-38.

[9] B. Fang, Y.Y. Wang, C.H. Leung, K.W. Tse, Y.Y. Tang, P.C.K. Kwok, Y.K. Wong, Offline signature verification by the analysis of cursive strokes, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 15, Issue: 4(2001) pp. 659-673.

[10] S. Inglis, I.H. Witten, Compression-based Template Matching, in proc: *IEEE Data Compression Conference* (1994), pp. 106-115.

[11] M.K. Kalera, S. Srihari, A. Xu, Offline signature verification and identification using distance statistics, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 18, 7(2004) pp. 1339-1360.

[12] N.S. Kamel, S. Sayeed, SVD-based signature verification technique using data glove, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 22, 3(2008) pp. 431-443.

[13] J. Wen, B. Fang, Y.Y. Tang, P.S.P. Wang, M. Cheng, T. Zhang, Combining EODH and Directional Gradient Density for offline signature verification, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 23, 6(2009) pp. 1161-1177.

[14] J. Bromley, J.W. Bentz, L. Bottou, I. Guyon, Y. Lecun, C. Moore, E. Sackinger, R. Shah, Signature Verificaiton using "SIAMESE" Time Delay neural network, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 7, 4(1993) pp. 669-688.

[15] I. Yoshimura, M. Yoshimura, off-line signature of Japanese signatures after elimination of background patters, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 8, 3(1994) pp. 693-708.

[16] W. Nelson, W. Turin, T. Hastie, Statistical method for on-line signature verification, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 8, 3(1994) pp. 749-773.

[17] B. Li, D. Zhang, K. Wang, Online signature verification by combining shape contexts and local features, *International Journal of Pattern Recognition and Artificial Intelligence*, Volume: 6, 3(2006) pp. 407-420.

[18] M.S. Akshoy, H. Mathkour, Signature verification using rules 3-ext inductive learning system, *International Journal of Physical Science*, Vol.6(18) (2011), pp. 4428-4434.

[19] S. Patil, S. Dewangan, Neural Network-based offline handwritten signature verification using Hu's moment invariant analysis, *International Journal of Engineering and Advanced Technology,* Vol.1(1) (2011), pp.73-79.

[20] L.Y. Tseng, T.H. Huang, An online Chinese signature verification scheme based on the ART1 neural network, in proc: *Int. Conf. Neural Networks,* vol.3 (1992), pp.624-630.

[21] P. Mautner, O. Rohlik, V. Matousek, J. Kempf, Signature verification using ART-2 neural network, in proc: *9th Int. Conf. Neural Information Processing*, vol.2 (2002), pp. 636- 639.

[22] T. Dash, S. Chattopadhyay, T. Nayak, "Handwritten Signature Verification using Adaptive Resonance Theory Type-2 (ART-2) Net". *Journal of Global Research in Computer Science* (2012) vol. 3 issue 8, pp. 21-25.

[23] T. Dash, T. Nayak, S. Chattopadhyay, "Offline Handwritten Signature Verification using Associative Memory Net". *International Journal of Advanced Research in Computer Engineering & Technology* (2012), vol. 1 issue 4, pp. 370-374.

[24] R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald, R. Menon, Parallel programming in OpenMP, *Morgan Kaufmann Publisher* (2001). ISBN 1-55860-671-8.