Accuracy Enhancement in Offline Signature Verification with The Use of Associative Memory

Anu Rathi
Assistant Professor
Dronacharya Group of institutions, Greater Noida

Amrita Ticku
Assistant Professor
Dronacharya Group of institutions, Greater Noida

Niti Gupta
Assistant Professor
Dronacharya Group of institutions, Greater Noida

ABSTRACT
In today's world every official document need the Hand written signature. To verify that the signature done by the person is accurate and not forged, human brain stores the pattern, data and everything so that they can recognize the pattern or data if again the same thing is shown. But the human mind receives the signal from eye which cannot catch the small variation in the pattern so it becomes a big problem for the legal issues. Associative memory is used for the verification of offline signature is used. In this paper we extend the algorithm for the memory of the system and how it will check the correct signature and forged signature .The Associative Memory Net (AMN) in correctly detecting forged signatures, fast. Here, the cost functions are handled with detail parametric studies and parallel processing using OpenMP. These algorithms are trained with the original or genuine signature and tested with a sample of ten very similar-looking forged signatures. The study concludes that AMN detects forgery with accuracy 94.3%, which is comparable to other methods cited in this paper.

Keywords
Signature verification, Offline, Bi-objective optimization, Associative Memory Net, OpenMP.

1. INTRODUCTION

The aim of off-line signature verification is to decide, whether a signature originates from a given signer based on the scanned image of the signature and a few images of the original signatures of the signer. Unlike on-line signature verification, which requires special acquisition hardware and setup, off-line signature verification can be performed after the normal signing process, and is thereby less intrusive and more user friendly. On the other hand, important information like velocity, pressure, up- and down strokes is partially lost. In the past decade a bunch of solutions has been introduced, to overcome the limitations of off-line signature verification and to compensate for the loss of accuracy. However when tested against skilled forgeries, even the best systems deliver worse equal error rates than 5%, in contrast with a human expert, who is able to do the distinction with an error rate of 1%. To break this barrier it is essential to identify, understand and compensate for the different sources of error in the algorithms.

Neural Network (NN) learns patterns by examples or observations [1].Learning can be 'supervised' or 'unsupervised'. Adaptive Linear Net (ADALIN), Multiple ADALIN (MADALIN), Perceptron Network, Radial Basis Function Net (RBFN) etc. are some important examples of supervised learning methods. 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. Associative Memory Net (AMN) updates its knowledge base through the concept of supervised learning. Depending on target pattern, AMN can be Auto-AMN or Hetero-AMN. In the former type, target is similar to the training pattern, while it is different in case of its counterpart. Typical signature verification approaches consist of 3 main phases. First they extract some features from the images of signatures, then they compare them and finally, they use some kind of classifier to decide whether a given signature is an original or a forgery. In this paper we are implementing an algorithm based on the principle of Auto-AMN. The verification of signatures is performed offline.

As signature verification is a bi-objective optimization problem where lowest error should be expected with least time, 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.

Manuscript received March 5, 2014
Manuscript revised March 20, 2014

2. Related work

We have reviewed various traditional and soft computing-based methods, used for signature verification. A method of signature verification using ART-1 was developed by [1] which resulted in an accuracy of 99.9% in case of skilled forgeries. This method was based on Training and testing of the network. A Bayesian network representation has been proposed by Xiao and Leedham, (2002). They proposed a decision tree like network, where each node of the tree computes the conditional probability as the chance of matching [2].In the year 2005, Kholmatov and Yanikoglu presented an online signature recognition scheme using Dynamic Time Warping (DTW) and three dimensional feature vectors. The overall accuracy rate of the method was 98.6%. The authors have used and [4]. Displacement extraction method has been proposed for offline signature verification. 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% [3].

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%.

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% [6].Wavelet thinning features were used for offline signature verification using Matching Algorithm. Similarity measurement was evaluated using Euclidean distance of all found corresponding feature points. The accuracy in this case was 81.4% [7].

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 [8].

The method using distance statistics yielded an accuracy of 78% for verification and 93% for signature identification [9].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% [10].

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: Take all signatures in pixel format.

Step-3: Implementation of AMN in 'C' language

Step-4: Training of the networks with genuine signatures.

Step-5: Set the genuine signature as the target

Step-5: If the signature is not matching then give the output. If the targeted output is equal to obtained output
(i) The signature is matching completely

Step 6: when the targeted output is not equal to obtain output then

(i) The signature is not matching completely.

Step-5: Testing of the developed AMN

Step-1: Collection of Signature samples

Original signature is produced at first and then forged signature samples are collected from different persons at different times . Each person has been given training for (10 days each) to enable copying the original signature with almost no visually detectable mistakes. All signatures are then saved into BMP files of size: 200×63dpi with Bit depth as 4. The signature templates are shown in Appendix-I.

Step-2: Feature Extraction

A user-defined image function in C-Program is used for extraction of pixels from the BMP files [11]. The method of feature extraction is given in Figure-1.

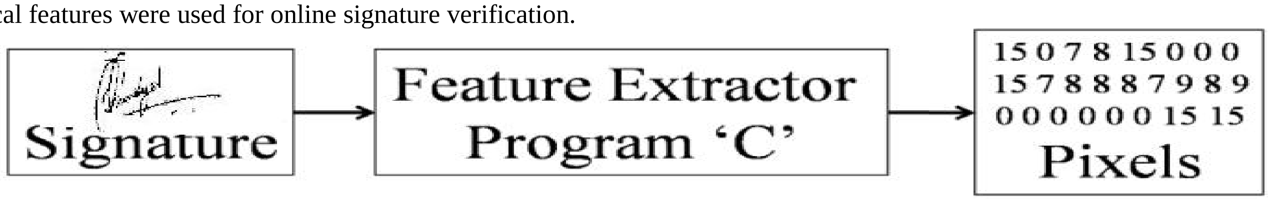

Fig1. Feature Extraction using our C-program.

3.1. Serial Implementation of the Network

Our developed algorithm for the approach and the training of the AMN network is given below. It will be wise to mention that training of the AMN had been done with the genuine sample only.

INITIALIZE weight (W) to 0

INPUT the original sign to the first layer of AMN.

Apply the Linear Transfer Function here and give the final output.

In LTF the input =output (for original signature)

FOR i=1 to n

DO

FOR j=1 to n

DO

Calculate the weight as

Wij (new) = Wij (old) + INPUT i× TARGET j

END //end of for loop on j

END //end of for loop on i

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 (Yin j > 0) Yj =+1;

ELSE

Yj=-1;

END //end of for loop on j

END //end of FOR loop on i

If (the TS = FS)

Then the signature is completely matched.

Else

The signature not matched.

3.2. Parallel Implementation of the networks

We developed a parallel version of all the above algorithms using OpenMP (www.openmp.org). By doing this, three major things could be achieved;

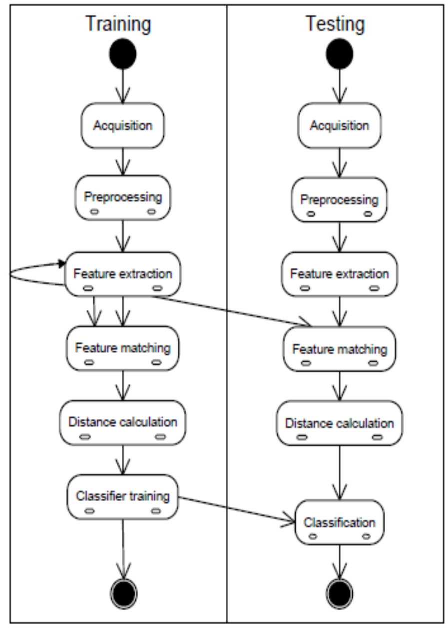

Figure2. Generalized view of an offline signature verification system.

i. Reduction in computation time,
ii. Utilization of all the processor of the system
iii. Inherent parallelism property of NN could be used.

3.3. System architecture specification

The implementation (both Serial and parallel) are carried out in a system having Intel Quad Core Processor with 4GB RAM having a processor speed of 2.10GHz. The operating system used is Linux (Ubuntu Version 11.10). It is mentioned that the package used for parallel processing is the OpenMP 3.0.

Step-4: Testing of the developed AMN

The developed and trained AMN network was then trained with 12 samples of Forged signatures (Refer Fig.2). 4.1. Calculation of mismatch The numbers of YJ which are -1 are counted (count) (see Section 3.1). The mismatch percentage can be calculated using the following Equation-1. Mismatch = [count/Total Number of Pixels] × 100 (1) 4.2. 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 25% to avoid acceptance of skilled forgeries. It is important to note that, such threshold setting must be situation specific and the choice of the administrator/user [13].

4. RESULTS AND DISCUSSIONS

The contribution of our work can be outlined as; development of an algorithm for signature verification scheme using AMN neural network. It is wise to note that all the implementation is carried out using both serial and parallel processing techniques. In this paper, study on signature area and its affect on computation time is provided.

A. 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.

B. Mismatch results of AMN Network

The result in Table-1 gives the performance of our algorithm for signature verification using AMN.

Table-1 Result of execution of Test cases

Test Case (Original Vs.)	**Mismatch**	**Decision (%)**	Computation Time (seconds) Serial	Parallel
Original	21.31	Accept	6.15	1.98
Forged1	29.29	Reject	7.9	2.44
Forged2	28.92	Reject	8.8	2.67
Forged3	29.86	Reject	12.06	3.38
Forged4	28.21	Reject	7.13	1.80
Forged5	28.44	Reject	10.24	2.87
Forged6	28.67	Reject	10.85	2.10
Forged7	29.65	Reject	11.46	3.02
Forged8	29.08	Reject	12.08	4.33
Forged9	31.02	Reject	12.69	4.54
Forged10	29.09	Reject	6.304	2.00
Forged11	21.57	Accept	7.91	2.89
Forged12	22.09	Accept	14.52	4.83

Looking at the mismatch results of AMN from Table-1, it can be said that the network is performing better for all the forged signatures (Forged 1-10). But the network is rejecting the original signature itself (See Test case-1 of Table-1). The network is also accepting two of the forged signatures, Forged-11 and Forged-12. The happened due to setting of the mismatch threshold to 25%. However, the average time taken by the serial algorithm is 9.85 seconds where as the parallel algorithm takes only 2.98 seconds.

Figure-3 Accuracy achieve by the algorithm for forged signatures

C. Acceptance accuracy of each forged signature

A plot has been given in Figure-3 to see which one of the forged signatures sample is getting verified with highest accuracy. Overall accuracy can be calculated as the number of correctly recognized sample test cases to the total number of test cases multiplied by 100. In our study, the accuracy is found to be 94.3%.

D. Forged Signature versus Computation time plot

A plot is given in Figure-4 showing CPU utilization time for AMN to recognize each of the forged signature samples.

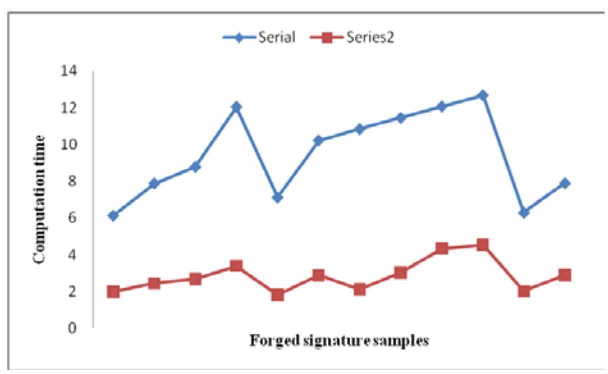

Figure-4 Time taken by algorithm to detect each forged signatures

Table-2 Comparative analysis of different methods of signature verification

Method	Year	Accuracy
ART-1[1]	2012	99.9%
Displacement Extraction Method[3]	2002	75%
DTW and 3D feature vector[4]	2005	98.6%
GA and Optimal Function[5]	2004	99.9
Co-operative NN[6]	1994	96%
Matching Algorithm[7]	2007	81.4%
Hidden Markov Model[8]	1994	88.9%
Distance statistics[9]	2004	78%
DTW[10]	2006	93.23
Associative Memory Net	2012	92.3%
AMN enhancement algorithm	2013	**94.3%**

5. Conclusion & Future Scope

Automatic signature verification is a very attractive field of research from both scientific and commercial points of view. In this paper, we proposed an extended version of adaptive algorithm based on neural network techniques for offline hand-written signature verification. All these methods have been optimized with respect to accuracy and computation time.

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. There are two versions of the algorithm, serial and parallel. The study revealed that our algorithm takes an average of 9.58 seconds in serial and 2.98 seconds in parallel to give a detection accuracy of 94.3%. The error threshold is set to 30 % in this work to make decision. It is important to mention that in this work we have tested our algorithms on a sample of only ten 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 tool links.

:

Figure-5 List of Original and forged signatures

6. REFERENCES

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

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

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

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

[5] Romo J.C.M., Silva R.A., 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] Cardot H., Revenu M., Victorri B., Revillet M.J., 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.

[7] Fang B., You X., Chen W.S., Tang Y.Y., 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.

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

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

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

[11] Dash, T., Nayak T. A Java Based Tool for Basic Image Processing and Edge Detection, Journal of Global Research in Computer Science, Volume 3, Issue 5 (2012) pp. 57-60.

[12] [Chandra R., Dagum L., Kohr D., Maydan D., McDonald J., Menon R., Parallel programming in OpenMP, Morgan Kaufmann Publisher (2001).

[13] Chattopadhyay S., Banerjee S., Rabhi F.A, Acharya R. U. A Case-based Reasoning System for Complex Medical Diagnoses. Expert Systems: the Journal of Knowledge Engineering (2012); DOI: 10.1111/j.1468-0394.2012.00618.x (in press).

[14] Tirtharaj Dash, Tanistha Nayak, Subhagata Chattopadhyay, “Offline Handwritten Signature Verification using Associative Memory Net” International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 4, June 2012