On Analysis of Multi-dimensional Features for Signature Verification

Jalal Mahmud* and Chowdhury Mofizur Rahman**
**Department of Computer Science*
Stony Brook University, NY 11794 USA.
*** Department of Computer Science*
United International University, Dhaka Bangladesh

*E-mail: *jmahmud@cs.sunysb.edu,**cmr@uiu.ac.bd*

Abstract

This paper aims to verify offline signatures using improved feature analysis and artificial neural network. Feature analyzer can reduce the large domain of feature space and extract invariable information. We incorporated different features from multi-dimensional feature analysis perspective. For verification from extracted features, we used neural network classifier. Instead of using feed forward neural network, multiple feed forward neural networks are used which are trained in the form of ensemble. Using such ensemble makes the system more general than a regular single neural network based system. Use of resilient back propagation for each neural network training, provides faster recognition. Using cross validation techniques, we performed significant amount of testing. Experimental evaluation of the signature verifier is reported.

1. Introduction

From the viewpoint of pattern recognition, the task of signature verification is to judge whether an input signature is a genuine signature or a forgery by comparing it with collected signature samples. The proposed system of signature verification works only for offline signatures to reduce the complexity of the system architecture and to avoid the need of specialized hardware for collection of online signatures. There has been a long line of works in automated signature verification [1,2,3]. Still the verification accuracy of the present signature verification systems is not very impressive and the systems are not very efficient in the presence of noise for offline signature verification. For the development of signature verification system, useful and efficient feature extraction [4,5,6] is a crucial step. A signature pattern can have a large number of measurable attributes, all of which may not be necessary for uniquely identifying it from other patterns in a particular domain of classification problem using a chosen classifier. Good features enhance within-class pattern similarity and between class pattern dissimilarity. Therefore feature extraction [7,8,9] is the most challenging part for pattern recognition problems. Selection of a particular class of feature vector varies from problem to problem. The choice of features to represent the patterns affects several aspects of the pattern recognition problem such as accuracy, required learning time and necessary number of samples.

For feature extraction from signature images, quad tree representation was incorporated with density analysis, moment analysis and structural analysis. For classification purposes, artificial neural network ensemble [10,11,12] have been used. The reason for using artificial neural network ensemble is to enhance generalization ability of the system. In order to get a fast learner, resilient back propagation algorithm has been incorporated. Extensive training and testing have been performed using 10 fold cross validation technique [13] that also proves the effectiveness of the system.

In the next section we describe preprocessing steps involved for signature verification system (SVS). In Section 3, we describe feature analysis of the signature images. In Section 4, we focus on classification and verification of the signature images. We performed experimental evaluation, which is described in section 5. Finally, we discuss limitation and future work in section 6.

2. Processing Signature Images

In the present system signature images have been obtained by optical scanning of the signature images on the plain paper. Preprocessing steps include size normalization, noise elimination and elimination of redundant information as far as possible. For digitization purpose, signatures are written on papers and acquired as binary images. Scaling of the signature images was performed so that size invariant recognition can be possible. The signatures that are scaled using an efficient scaling algorithm [14] converted to standard size, which was 64 x 64 for the system. A filtering function was used to remove the noises in the image. Filtering function worked like a majority function that replaced each pixel by its majority function.

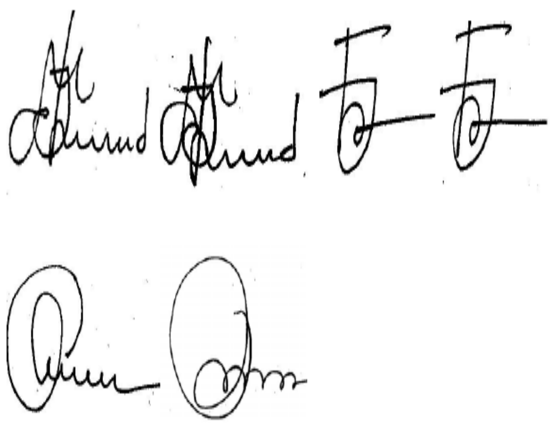

Fig 1: Some sample signatures captured by the scanner

Preprocessed signature image was divided into some regions to have a quad tree representation. Each image was divided into four regions depending on the center of mass and bounded rectangle of the image. Such a division yielded quad tree of depth one. Similar procedure might be applied to get quad tree of higher depth. The mathematical definition of center of mass:

$$X_{CM} = \sum_{j=1}^{N} X_j / N \;\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (1)$$

$$Y_{CM} = \sum_{j=1}^{N} Y_j / N \;\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

Here N = Number of Black pixels in the image,
X_j = x Coordinate of Black pixel in the image,
Y_j = y Coordinate of Black pixel in the image.

The mathematical definition of R_1, R_2, R_3 and R_4 are,

$R_1 = (X_{min} \ldots\ldots\ldots X_{CM}, Y_{min} \ldots\ldots \ldots Y_{CM})$
$R_2 = (X_{min} \ldots\ldots\ldots X_{CM}, Y_{CM+1} \ldots\ldots\ldots Y_{max})$
$R_3 = (X_{CM+1} \ldots\ldots X_{max}, Y_{CM+1} \ldots\ldots Y_{max})$
$R_4 = (X_{CM+1} \ldots\ldots X_{max}, Y_{min} \ldots\ldots \ldots Y_{CM})$

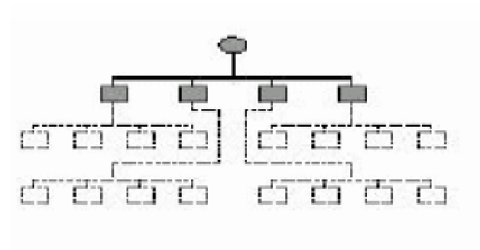

Fig 2. Quad tree representation

3. Feature Analysis

The size of the feature set is important in order to avoid a phenomenon called the dimensionality problem. In the proposed method, several types of features are extracted from the scaled input signature image, these includes density features, moment features and structural features. The overall feature extraction may be divided in some phases, which are described below.

3.1. Density Based Features

Pixel density in each of the regions from quad tree of depth one and higher were calculated to get density features. Density feature in quad tree region i is denoted by $D_{i,}$ where,

$$D_i = \sum_j (I(\beta_i)) / (N_i) \;\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3)$$

Here, β_j is the jth black pixel,
I is an intensity function.
N_i is the total number of pixels in region i.

Density features were calculated from quad tree of depth one and two. So 4 density features from quad tree depth one and 16 density features from quad tree of depth two were collected. Normalized density features generated 20 feature vectors.

3.2. Moment Features

Using the histogram of the contour [8,9], moment features may be effectively calculated. 8 moment features were obtained, using the signature image histogram. Ratios of the length of the sum of contour segments, which are present within a sub-image to the total contour perimeter, generated the Moment features. Moment feature for contour component j is denoted by μ_j, where

$$\mu_j = \left(\sum_{i=1}^{n} \prod_{k=1}^{n_i} (D_i, \beta(k))\right) / L_j \quad \ldots\ldots(4)$$

Here, L_j = Total length of contour component j. D_i = Pixel density of region i. n_i = Number of segments in region i. $\beta(k)$ is a function that gives the chain coded value of contour segment k. For the present analysis, moment features were calculated from quad tree of depth one and two, thus generating 8 feature vectors from depth one and 32 feature vectors from depth two regions. Therefore, we used 40 such moment features.

3.3. Structural Features

Structural features from the signature image have been calculated for gaining the local information from the regions. Structural features conveys the following information,

3.3.1. Coarse structural features. A count of the number pixels in the thinned segment is obtained during segment tracing. This is normalized with respect to the area of the bounding box of the signature image. With the information of the number of pixels in the contour in a particular region, it is possible to gain an insight into the space filling property. For 4 Quad tree regions, there are 4 Space filling features that may be denoted as S_i,

$$\text{Where } S_i = \sum_j F(\alpha_{j,i})/A \quad \ldots\ldots(5)$$

$\alpha_{j,i}$ = Number of active pixels in the boundary of the image in region i, F is an intensity function. And A is the area of the bounding box. Therefore we calculated 4 space filling features.

3.3.2. Directional features. Stated previously, critical points are calculated from the contour analysis of the image. The length of each segment is detected and the directivity of each segment indicates directional property of the image. Further analysis on directional property yields directional strength measure that effectively generates 4 new structural features denoted as θ_1, θ_2, θ_3 and θ_4. The mathematical definition of the directional features denoted by θ are defined here:

$$\theta_i = \sum_j (\Psi_{j,i}) / m_j,\ m \le n \quad \ldots\ldots(6)$$

$\Psi_{j,i}$ denotes the directivity of jth segment in the ith sub image. Here m_j denotes the total number of segments in sub image i. n_j denotes the total number of chaining elements in region i.

From the definition of these structural features, a set of modified structural features may be computed, which actually measure the directional strength of a region. Directional strength measures the weight value of a piece wise linear contour segment. Mathematical definition of directional strength of a particular contour segment can be given as:

$$\partial_i = \mu \prod_{j=1}^{m_i} L_i f(L_i, \lambda_{i,,j}) \quad \ldots\ldots(7)$$

∂i is the directional strength of segment i.

m_i is the number of neighbors of segment i. Here, μ is the momentum constant whose value ranges between 0.001 to 0.999.For this particular experiment the value of μ was set to 0.5. L_i is the length of particular segment i, measured in number of pixels. $\lambda_{i,,j}$ is the strength of jth element in ith segment. For this specific experiment $\lambda_{i,,j}$ equals the number of active neighbors of jth element of ith segment.

f is an importance function which is defined as :

$f(x,0) = 1$,

$f(x,y) = y$, when $x > Z_0$, where Z_0 is a threshold value.

$f(x,y) = y^2 - y + Z_0$, otherwise.

For the present analysis, 4 coarse structural features and 4 directional features were calculated for quad tree region one. Their normalized values generated 8 structural feature vectors.

4. Building the Verifier

Signature data is highly non-linear in nature as a result of varying styles. We used neural network [15], which is a non-linear classifier for verification purpose.

4.1 Neural Network

The general architecture of the neural network is shown in the figure 3. The network is arranged in multiple layers, each layer containing fixed number of nodes for the specific problem. The nodes of the input layer are called input nodes, output layers are called output nodes and the intermediate layers are called hidden nodes where all the intermediate layers are designated as hidden layers. Each input node is connected to each hidden node, and each hidden node is connected to each output node. There is a weight associated with each path between nodes. There is also a weighted bias feeding into each hidden and output node. Therefore the input to each hidden node is the sum of all on the input nodes times the weight along the path plus the weighted bias to that node. The output from that hidden node is then determined by passing the input through an activation function. In figure 3, the activation function is a bipolar (or tan) sigmoid function.

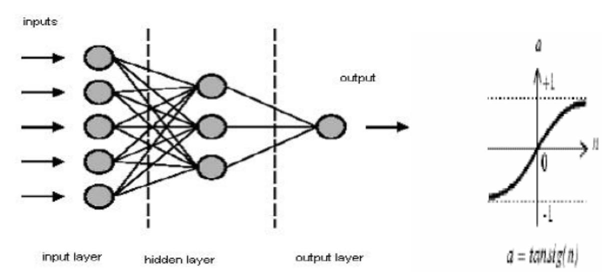

Fig 3: General Neural Network Architecture and Activation Function

4.2. Network Size

There is no rigorous rule about choice regarding the size of the network. The size for which convergence speed is higher can be used for training and recognition purpose. The learning constant of the neural network is usually kept small for faster convergence. Generally size of the input layer is made equal to the dimension of feature vectors and size of output layer is made equal to the dimension of output class. We used 20 density features, 40 moment features and 8 structural features, total of 68 features. Therefore for the present application, size of the input layer was fixed at 68, but we varied the size of hidden layer. Size of output layer was made equal to the number of subjects from whom we collected the signatures.

4.3. Training Phase

The feed forward back propagation network undergoes supervised training, with a finite number of pattern pairs consisting of an input pattern and a desired output pattern. The network was trained by modifying the weights between the layers. For each training iteration (epoch), the error was calculated by taking the difference between the output and the expected output. In the usual back propagation algorithm, the gradient (calculated using the derivative of the activation function) was used to determine the change in the weights. However, especially in the second layer of back propagation, the result of the derivative of the activation function can produce a very small number, so there will be a very small change in the weights. In resilient back propagation [18], only the sign of the gradient is used. The weight is then changed by one of two constant values depending on the sign of the gradient. This allows a net to learn much more quickly. Therefore to get quick convergence we implemented resilient back propagation algorithm for learning the weights of the neural network.

4.4. Use of artificial neural network ensemble

Instead of using single neural network, an artificial neural network ensemble [10,11,12] was used for training and testing purposes. Say the training data set is $S = \{(x_1, y_1), (x_2, y_2),\ldots,(x_n, y_n)\}$, where x_i and y_i are the feature vector and the expected class label of the i-th training instance, respectively. An artificial neural network ensemble was trained with S. An artificial neural network ensemble was built in two steps, that is, generating component artificial neural networks and then combining their predictions. As for generating component networks, Bagging [16] and Boosting [17] are prevailing approaches. Bagging generated multiple training data sets from the original training data set and then trained a component network using each of those training data sets. Boosting generated a series of component networks whose training data sets are determined by the performance of the former networks. Training instances that are wrongly predicted by the former networks will play more important roles in the training of the later networks. As for combining component predictions, voting is prevailing for classification. Voting regards the class label receiving the most number of votes as the final output of the ensemble. For the present system, voting was used for component prediction combination. T bootstrap samples S_1, S_2, …, S_T were generated from the original training data set and a component artificial neural network N_t was trained using each S_t, an ensemble N^* is built from N_1, N_2, …, N_T whose output is the class label receiving the most number of votes. Since artificial neural network ensembles usually have strong generalization ability, some noise was depressed by the process of N^*.

4.4 Recognition

In the recognition phase of the network a single iteration was enough to give the confidence value for each class of the character set. The value obtained from the output layer of the neural network, which closes to 1 implies the presence of that class.

5. Experimental Evaluation

For the signature verification system, signatures were collected from different persons. For each person, 10 sample signatures are collected. 30 persons participated in the experiment. The system was trained using 10 original signatures and 5 forgeries generated for each person. Original signature served as positive examples and forgeries as well as other signatures in the training data acted as negative examples. The number of persons taking part in the experiment were varied to observe the verification accuracy at various levels. The samples were divided into two parts, one for training phase and one for recognition phase to ensure cross validation. For the present system, 10 fold cross validation was used. For 10 fold cross validation [13], at any one time, 90% of the data was used for training and the performance was tested on the remaining 10%. 10-fold cross validation was performed in each case study. In each fold, an artificial neural-network ensemble comprising sixteen individual networks was generated via Bagging [16]. The training data sets of the neural networks are bootstrap [19] sampled from the training data set of the fold. During the training process, the generalization error of the network was estimated in each epoch on its validation data set. If the validation error did not change in consecutive six epochs, the training of the network was terminated in order to avoid over fitting. For each subject taking part in the experiment, we computed two metrics precision[1] and recall[2]. We varied number of hidden neuron in the hidden layer of the neural network and observed the resultant networks performance. We also introduced random noise (deviation) in the signatures and measured the resultant accuracy of recognition. Our experimentation showed that verification accuracy (precision and recall) ranged from 78% to 94%. Number of hidden neuron of each of the neural network was varied and corresponding mean squared error was measured. Minimum error was found when number of hidden neuron was 60.

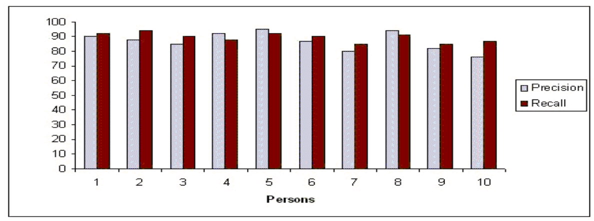

Fig 4. Precision and Recall Characteristics

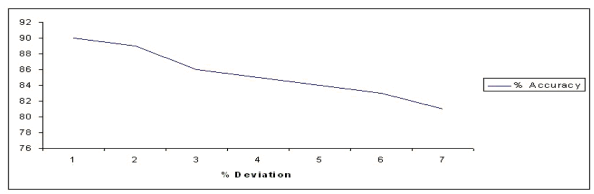

Fig 6. Accuracy Variation with Deviation Level

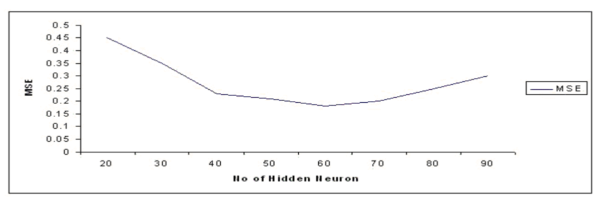

Fig 7. Error variation with Hidden Neuron

6. Discussion and Future Work

A new approach for signature verification system has been presented and implemented. Analysis of features from different dimension enables the system to have a set of feature vectors which are noise invariant. Present system of signature verification also employs efficient non linear classifier like neural network. Speed of convergence has been effectively enhanced by resilient back propagation algorithm. Significant amount of training and testing also performed and cross validation approach during training and testing enables the system to have robust recognition which is impressive as seen from the experimental result presented. Signature of same

---

[1] Precision is the percentage of correctly identified signatures over total (correctly and incorrectly) identified signatures.
[2] Recall is the percentage of true signatures correctly identified.

person is varied highly for various factors like pen pressure, emotion, environmental factors. This is why an offline signature verification system may not be able to verify signatures in all circumstances. To improve the rate of verification, the system should be online that involves extraction of stylistic features that are changed while writing on the paper. Even for the offline signature verification system, to accelerate the verification rate, features that are capable of describing the individual strokes, junctions and holes should be extracted. Incorporation of such features is a work in progress.

7. References

[1] I. Evett and R. N. Totty, "A study of the variation in the dimensions of genuine signatures," *J. Forensic Sci. Soc.*, Vol. 25, pp. 207–215, 1985.

[2] T.Hastie, E.Kishon, M.Clark, and J.Fan, "A model for signature verification," in *Proc. of IEEE Int Conf. on Syst,ManCybern.*,Vol.1,Charlottesville,VA,Oct.1991,pp.191-196, 1991.

[3] F. Leclerc and R. Plamondon, "Automatic signature verification: The state of the art 1989-1993," *Int. J.Patt. Recognit. And Artificial Intell.*,Vol. 8, No. 3, pp. 643–660, June 1994.

[4] D.Rutovitz, "*Pattern Recognition*", Journal of Royal Statistical Society, Vol. 129,Series A, pp. 504-530, (1966).

[5] T. Pavlidis, "*A Vectorizer and Feature Extraction for Document Recognition*", Computer Vision, Graphics and Image Processing, Vol. 35, pp. 111-127, (1986).

[6] T. Pavlidis, "*Algorithms for Graphics and Image Processing*", Springer-Verlag,Berlin, (1982).

[7] S. Yokoi, J.I. Toriwaki, T. Fukumura, "*An Analysis of Topological Properties of Digitized Binary Pictures Using Local Features*", Journal of Computer Graphics and Image Processing, Vol. 4, pp. 63-73, (1975).

[8] C.C.Lue and J.G Dunhaum, "Highly Efficient Coding schemes for contour lines based on Chain Code Representation", IEEE Trans, Comm Vol. 39, No. 10,pp. 1511-1514., 1991.

[9] H.Freeman ,"Computer Processing of Line Drawing Images, Computing Surveys",Vol. 6, No. 1,pp. 57- 97,1974.

[10] L. K. Hansen and P. Salamon, "*Neural network ensembles*," *IEEE Trans.Pattern Analysis and Machine Intelligence*, vol.12, no.10, pp.993-1001,1990.

[11] L. K. Hansen, L. Liisberg, and P. Salamon, "Ensemble methods for handwritten digit recognition," in *Proc. IEEE Workshop on Neural Networks for Signal Processing*, Helsingoer, Denmark, 1992, pp.333-342.

[12] Cunningham, J. Crney, and S. Jacob, "Stability problems with artificial neural networks and the ensemble solution," *Artificial Intelligence in Medicine*, vol. 20, no.3, pp.217-225, 2000.

[13] M. Stone, Cross-Validatory Choice and Assessment of Statistical Predictions, Journal of the Royal Statistical Society. **36** (1), 111-147, 1974.

[14] Suman Kumar Nath and Muhammad Mashroor Ali, An Efficient Object Scaling Algorithm for raster device", raphics and Image Processing, NCCIS , 1997.

[15] C. M. Bishop, Neural networks for pattern recognition, Clarendon Press, Oxford, 1995.

[16] L. Breiman, "Bagging predictors," *Machine Learning*, vol.24, no.2, pp.123-140, 1996.

[17] H. Drucker, R. Schapire, and P. Simard, "Improving performance in neural networks using a boosting algorithm," in *Advances in Neural Information Processing Systems 5*, S.J. Hanson, J. D. Cowan, and C. L.Giles, Eds. San Mateo, CA: Morgan Kaufmann, 1993, pp.42-49.

[18] T. P. Vogl, J. K. Mangis, A. K. Rigler, W. T. Zink, and D. L. Alkon, Accelerating the convergence of the back-propagation method, Biological Cybernetics, **59**, 257-263, 1988.

[19] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. New York: Chapman & Hall, 1993.