APPLICATION OF EFUNN FOR THE CLASSIFICATION OF HANDWRITTEN DIGITS

G. S. Ng, T Murali, D. Shi and A. Wahab

Centre for Computational Intelligence, School of Computer Engineering,
Nanyang Technological University, Singapore 639798

asgsng@ntu.edu.sg

ABSTRACT
Handwritten digits classification has many useful applications. This has prompted decades of research into algorithms to produce an effective system of classifying handwritten images into text. Image processing and feature extraction play a large role in this process. An intelligent system is one, which is taught, and one, which uses this learning for classification effectively. The neuro-fuzzy model of Evolving Fuzzy Neural Network (EFuNN) is used for this purpose. This paper aims to analyse and obtain the optimal number of features that will produce the most effective classification using EFuNN.
Keywords: Neural networks, Evolving computation, Handwritten digit recognition, Image processing.

1. INTRODUCTION

At an early age, humans pick the skill of recognizing digits and letters. While this seems instinctive, it becomes a lot harder to understand how we are faced with the task of teaching a machine to perform the same task of recognizing digits. Machines are most familiar with data and the primary task would be to process the images of handwritten digits to data. This process would be led by processing the image. From the enhanced image, useful features are extracted. These features play an important role in teaching a system to classify. An appropriate model is used for the purpose of teaching and classification. For an effective classification by the system, this model has to satisfy several criteria.

The performance of a classifier depends on the interrelationship between sample sizes, number of features, and the classifier's complexity. This paper focuses on investigating the performance of the classifier by varying the sample size and the number of features. There are reasons to keep the dimension of the data representation, i.e. the number of features, as small as possible. These are cost in terms of memory consumption, time and classification accuracy. A limited feature set simplifies both the pattern representation and the classifiers that are built on the selected representation. As a result, the classifier will be faster and will use less memory. On the other hand, a reduction in the number of features may lead to a loss in the discrimination power and thereby the accuracy of the resulting recognition system.

In section 2, the process of image processing and feature extraction will be covered. The classifier used is a neuro fuzzy model called Evolving Fuzzy Neural Network (EFuNN) (Kasabov, 2001). The architecture of this model will be discussed in section 3. An analysis on the performance of EFuNN will be discussed in section 4. Finally, we report our findings of the performance of EFuNN on the task of handwritten digit classification.

2. IMAGE PROCESSING AND FEATURE EXTRACTION

Handwritten images have to go through stages of image processing before useful information can be extracted for classification purposes. Two image-processing procedures, edge enhancement and thinning are performed on the images to enable edge detection of the images. Edge detection reduces an object in an image into a thinned edge object that represents the contour of the object. The reason for using thinned edge image rather than the unprocessed image is that the thinned edge image reduces the amount of data that is needed to compute in the feature extraction process. After edge detection, feature extraction is performed. Feature extraction is an important task in the first step of the character recognition. Devijver and Kittler (1982) defined feature extraction as the problem of "extracting from the raw data the information which is most relevant for classification purposes, in the sense of minimising the within-class pattern variability while enhancing the between-class pattern variability. KL transform is used as the feature extractor. The KL transform is a linear transform and corresponds to the projection of images onto the orthonormal eigenvectors of the covariance matrix of a large number of prototype images. The prototype images are representative of the types of images desired to be recognised by the recognition system, in this case, character images. The production of this transform is also known as principal component analysis (Johnson, R.A. & Wichern, D.W. 1992) (Andrews, 1971). The KL transform requires the computation of the covariance matrix and is followed by its diagonalisation to produce the eigenvectors (Press, W.H. et al., 1996). The resulting eigenvectors can be used as basis vectors for feature extraction. The first 16 elements are used as they are seen as the most significant from observation.

3. EVOLVING FUZZY NEURAL NETWORK

The Evolving Fuzzy Neural Network (EFuNN) model is different from other fuzzy neural network models despite structural similarities. EFuNN evolves according to the Evolving connectionist system (ECOS) principles (Kasabov, 1996) (Kasabov, 1998). ECOS evolves its structure and functionality over time through interaction with the environment. It emerges, evolves, develops, unfolds through learning and through changing its structure in order to better represent incoming data.

EFuNN is a five-layered fuzzy neural network (see Figure 1). Its nodes and connections are created or connected as data samples are presented. Layer 1 takes the input variables. At layer 2, each input variable is represented by a group of spatially arranged neurons to represent fuzzy quantisation. The task of this layer is to transfer the input values into membership degrees to which they belong to a membership function (MF). Thus, if an input variable fails to belong to any existing MF to a degree greater than a membership threshold, new neurons can be evolved.

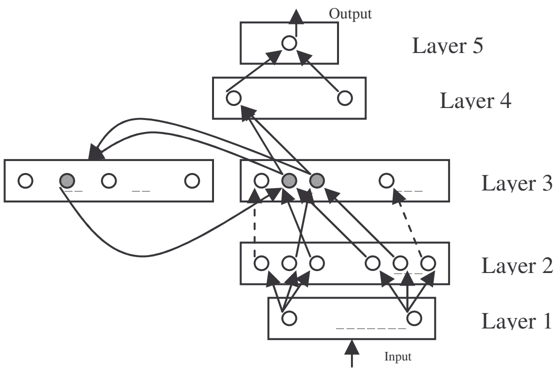


Figure 1: The five-layered structure of EfuNN.

Layer 3 contains rule nodes that evolve through learning. The rule nodes represent clusters of input-output data associations. Each rule node r is defined by two connection weights $\mathbf{w1}$ and $\mathbf{w}2$, i.e.

$$\mathbf{w1} = \mathbf{x} \text{ and } \mathbf{w2} = \mathbf{y} \tag{1}$$

where $\mathbf{x}$ is current fuzzy input vector and $\mathbf{y}$ is its corresponding output vector. $\mathbf{w1}$ is adjusted through supervised learning based on the output error and $\mathbf{w}2$ is adjusted based on similarity on the cluster.

Layer 4 represents the fuzzy quantisation of the output variables similar to layer 2. Using the weighted sum input function and a saturated linear activation function, the degree to which an output vector belongs to an output MF is calculated. At layer 5, a linear activation function is used to calculate the output variables (defuzzification).

The learning algorithm of EFuNN mainly consists of following steps: network initiation, inputs feeding forward, connections updating, parameters tuning, node aggregation, node pruning, and rule extraction.

4. EXPERIMENT ON HANDWRITTEN DIGITS CLASSIFICATION

In this experiment, further selection within the feature set has been made to analyse the performance of the classifier. As mentioned in earlier, the sample size and the number of features also determine the performance of the classifier. Thus, in the experiments, the sample size N and the number of features F are varied to determine the performance of EFuNN based on the following criteria.

1. Recall capability
2. Classification
3. Timing
4. Memory requirement
5. Root Mean Square Error (RMSE)

The data set consists of an N by F matrix whereby N ranges from 1 to 5000 and F varies from 1 to 16. Maximum 16 features are considered as they are observed to be discriminative enough. The notation F-1 means one feature is used, F-2 means two features are used and so on.

The data set is the result of image processing and feature extraction. Each row consists of F features that are the training values for a specific digit. These F values obtained from the KL transform vectors for a particular handwritten image of a digit. The first row in the matrix contains the training value for the digit '0' and the next row for digit '1' and so forth till digit '9'. Thus 10 rows have the training data for digits 0 to 9. Therefore, training data set consists of 500 different training data for each digit.

4.1. Recall Capability

Recall capability refers to the system's capability to classify accurately when it is tested with the same set of data it was trained with. Accuracy is measured on a scale from 0 to 1. In this experiment, EFuNN was trained with data of varying number of features F. For each F, N (i.e. sample size) was varied from 200 to 5000. Subsequently, the trained system is tested for recall capability using the same data.
From Figure 2, it can be seen that as the number of features increase, the accuracy increases. Increase in accuracy for an additional use of feature becomes minimal after 11 features. The detail can be seen in Table 1. Generally, with 6 or more features, the system performed at a high accuracy of 0.9 and above. Since, at recall, the training data and testing data are the same, the number of features does not make significant alterations to the accuracy. Greater number of features is more useful when the training and testing data are different. In such cases, the system should be adaptive to classify new input data. Then, greater number of features could be useful.

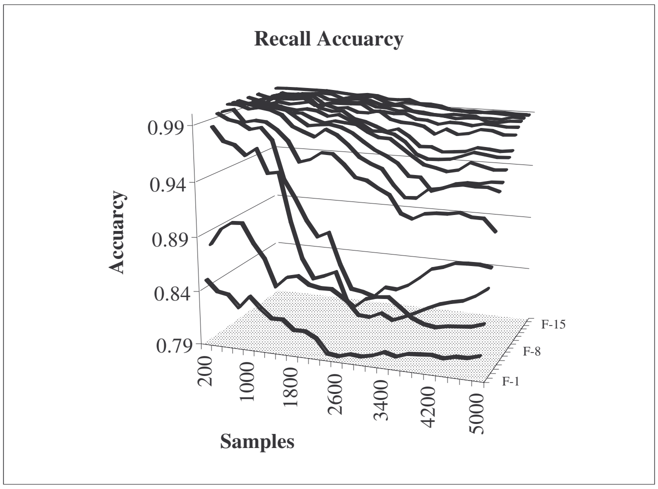


Figure 2: Recall capability with 16 features.

Table 1: Recall accuracy of selected features .

N \ F	6	7	8	9	10	11	12	13	14	15	16
1000	0.997	0.997	0.997	0.994	0.996	0.999	0.999	1	0.998	0.998	1
2000	0.983	0.986	0.987	0.991	0.993	0.994	0.996	0.996	0.997	0.997	0.999
3000	0.951	0.961	0.974	0.976	0.97933	0.984	0.989	0.991	0.989	0.994	0.993
4000	0.938	0.936	0.949	0.958	0.962	0.976	0.984	0.989	0.986	0.989	0.989
5000	0.939	0.945	0.954	0.963	0.967	0.979	0.984	0.987	0.988	0.989	0.988

4.2 Classification

The classification of the system is done using 60% of the data for training and 40% of the data for testing.

Figure 3 shows that with increasing number of features, the classification accuracy increases. The increase in accuracy is distinguishable at 6 features. Beyond this, change in accuracy is minimal regardless of the number of features used.

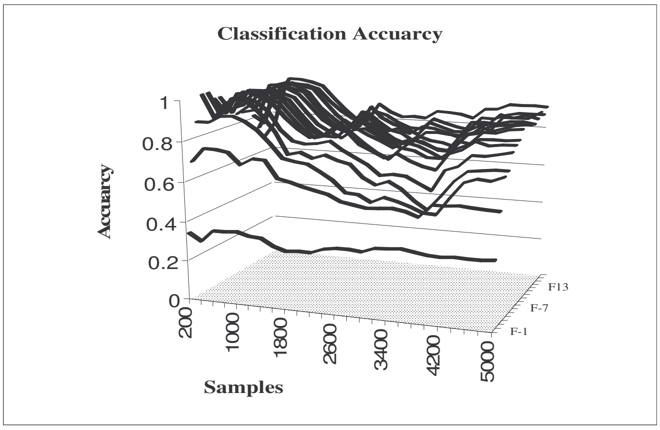


Figure 3: Classification accuracy with 16 features.

There appears to be a convergence after 6 or more features are used. Figure 3 shows that for small sample sizes (up to 2000 samples), the classification accuracy converges for various numbers of features. For subsequent larger samples, it is noticed that more features enhanced the classification capability. As it can be seen from Table 2, for samples greater than 3000, the system with 16 features outperformed the other systems. This is because, with larger samples, there are more test data (40% of N samples are used for testing). With more test data, the variability in data is likely to be higher. Then more features, contributing to greater training data will enable better classification.

Table 2: Classification accuracy of selected features.

N \ F	6	7	8	9	10	11	12	13	14	15	16
1000	0.983	0.98	0.983	0.97	0.97	0.985	0.976	0.983	0.978	0.973	0.983
2000	0.808	0.798	0.805	0.77	0.755	0.778	0.786	0.776	0.775	0.791	0.789
3000	0.716	0.7	0.745	0.759	0.72	0.733	0.776	0.724	0.750	0.77	0.819
4000	0.758	0.765	0.799	0.815	0.804	0.799	0.853	0.834	0.811	0.846	0.881
5000	0.819	0.832	0.874	0.897	0.877	0.866	0.859	0.876	0.900	0.879	0.909

4.3. Timing

4.3.1. Training Time

Quick training time is an important criterion of an effective classification system. Thus, it is useful to investigate the number of features on the duration of training. As it can be seen from Figure 4(a), as the number of features increase, the training time also increases. The numerical result is shown in Table 3. As the size of samples increases, the training time also increases.

4.3.2. Classification Time

Classification time refers to the time the system takes to classify digits when it's tested with different data. Figure 4(b) shows the classification times for system using different number of features at different sample sizes. The numerical result is shown in Table 4. Like training time, the classification times show linear relationship to the size of training data. As it can be seen from Figure 4(b), system trained with greater number of features, take longer time to classify digits. This is due to more rule nodes in the system. When there is greater number of rule nodes, system would require greater time finding the most appropriate rule to 'fire' for a given a set of inputs.

4.4. Memory Requirement

Table 5 shows the various system with number of memory nodes they possess. System using more features needs more rule nodes.

4.5. Root Mean Square Error

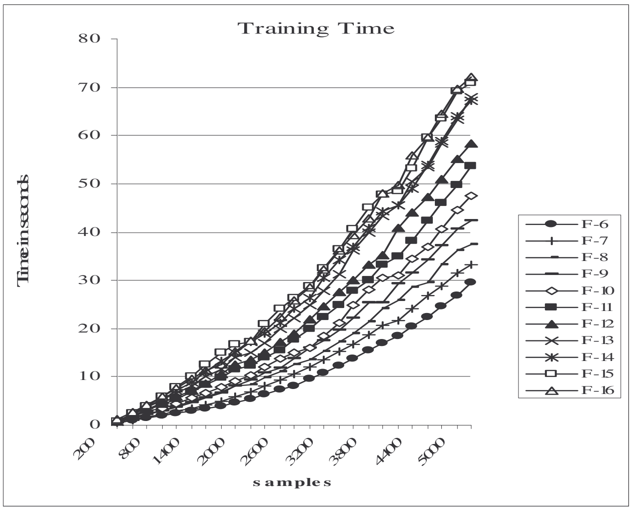
Root Mean Square Error (RMSE) is a good gauge of estimating the error rates of the two systems. RMSE is obtained during training as well as classification.

4.5.1. Training RMSE

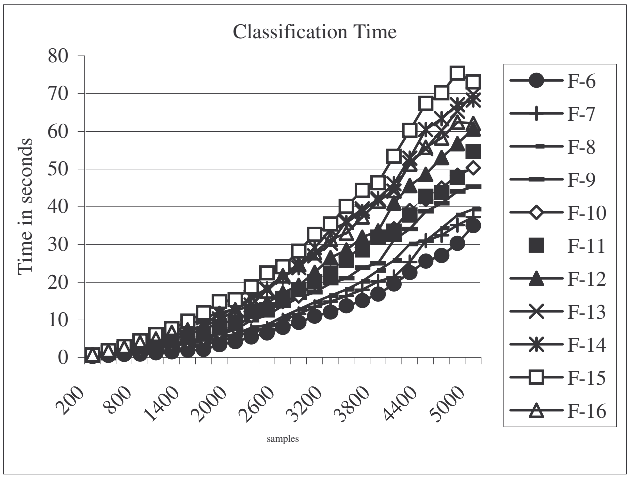
Training RMSE refers to difference between the sample training data and the rule nodes. System with greater number of features causes greater training error (see Figure 5(a)). This is because with greater number of features, the training data becomes larger. With more data, likelihood for error is greater.

4.5.2. Classification RMSE

Classification RMSE is calculated when the trained system is tested with sample test data. Although this is not as useful as training RMSE, it provides information as how effective the rule nodes are when tested with different data. From Figure 5(b), it's quite clear that during classification, the number of features does not affect error rates greatly.

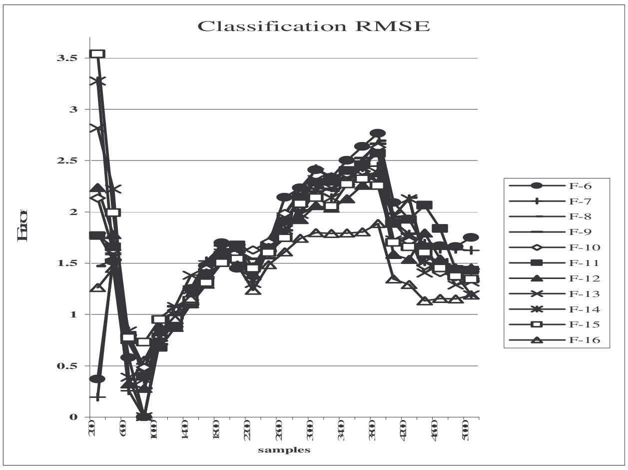


(a)

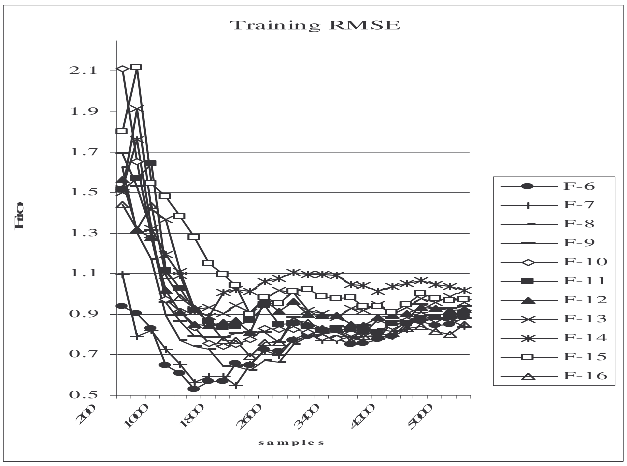


(b)

Figure 4: (a) Training time of selected features. (b) Classification time of selected features.



(a)



(b)

Figure 5: (a) Training RMSE of selected features. (b) Classification RMSE of selected features.

Table 3: Training time of selected features.

N \ F	6	7	8	9	10	11	12	13	14	15	16
1000	2.39	2.91	3.92	3.75	4.41	5.48	5.81	6.70	7.20	7.95	7.53
2000	5.41	6.92	8.42	9.23	10.36	12.23	13.62	15.06	17.25	17.33	17.49
3000	10.92	13.48	15.35	17.452	18.52	22.28	24.59	27.79	30.53	32.47	32.12
4000	18.53	21.73	25.89	29.25	31.08	35.03	40.87	45.53	45.47	48.47	49.84
5000	29.61	33.29	37.31	42.26	47.61	53.59	58.37	67.92	67.22	70.95	72.01

Table 4: Classification time of selected features.

N \ F	6	7	8	9	10	11	12	13	14	15	16
1000	1.25	1.61	2.39	2.61	2.92	3.59	3.64	4.5	5.67	6.08	4.97
2000	4.23	5.59	6.26	7.38	8.03	8.98	9.69	12.34	12.98	15.38	12.67
3000	10.95	13.45	14.67	17.29	19.94	20.03	22.61	26.95	29.16	32.62	27.79
4000	19.5	21.34	25.76	31.16	34.16	33.48	40.97	44.09	46.14	53.37	44.29
5000	34.95	37.31	39.34	45.25	50.29	54.56	60.59	69.78	68.28	73.06	62.11

Table 5: Memory requirement of selected features.

N \ F	6	7	8	9	10	11	12	13	14	15	16
1000	23	33	55	62	69	87	88	111	141	151	110
2000	48	67	76	91	100	111	120	154	162	192	135
3000	91	113	124	147	169	167	189	225	243	269	202
4000	124	136	164	200	218	210	256	275	287	329	242
5000	180	192	202	232	256	273	302	345	337	358	272

Table 6: Ranking of systems performance to various criteria.

Recall Capability	Classification	Memory Requirement	Classification Time	Classification Time	Training RMSE	Classification RMSE
F-15	F-16	F-6	F-6	F-6	F-7	F-16
F-16	F-15	F-7	F-7	F-7	F-6	F-12
F-14	F-14	F-8	F-8	F-8	F-8	F-13
F-13	F-13	F-9	F-9	F-9	F-9	F-14
F-12	F-12	F-10	F-10	F-10	F-16	F-15
F-11	F-9	F-11	F-11	F-11	F-10	F-10
F-10	F-8	F-12	F-12	F-12	F-11	F-11
F-9	F-11	F-16	F-13	F-16	F-12	F-7
F-8	F-10	F-13	F-14	F-13	F-13	F-9
F-7	F-7	F-14	F-15	F-14	F-14	F-8
F-6	F-6	F-15	F-16	F-15	F-15	F-6

5. CONCLUSION

Effective classification of handwritten digits using EFuNN was investigated and the result was studied. As mentioned earlier, effectiveness is measured by the performance of the system to certain criteria. These are recall and classification, memory requirement, timing and RMSE. At the stage of feature extraction, it's required to know the lower bound of the number of features that would be required to optimally meet the criteria. It was conclusive that system using 5 features or less performed inadequately. The summary of the results for system using 6 to 16 features is shown in Table 6. The table ranks system in descending order of recall and classification accuracy. More features give higher classification accuracy. Memory consumption is ranked in ascending order of number of rule nodes in the system. This means system using less number of features has less number of rule nodes. Training and classification time are ranked in ascending order of time spent. Training and classification RMSE is ranked in ascending order of errors. Simply, for all the criteria, higher rank indicates greater performance. Of the listed set of criteria, classification, memory consumption and timing (training and classification) has precedence over the others. From analyzing the results we can deduce that the trend indicates that system using greater number of features ensure greater classification accuracy, but they consume more memory and require more time to train and classify. Ultimately, EFuNN using 9 features satisfies the criteria of an effective classification system. Thus, it can be concluded that 9 most discriminative features from the process of PCA under feature extraction ought to be enough for training and classification of the system.

REFERENCES

Andrews, H.C. (1971) "Multidimensional rotations in feature selection",. *IEEE Transactions on Computers*, 1045-1051.

Devijver, P.A. & Kittler, J. (1982) *Pattern Recognition: A Statistical Approach*, Prentice Hall.

Johnson, R.A. & Wichern, D.W. (1992) *Applied multivariate statistical analysis,* Englewood Cliffs, Prentice Hall.

Kasabov, N. (1996) *Foundation of neural networks, fuzzy systems and knowledge engineering*, 1st ed. Cambridge, MA, MIT Press.

Kasabov, N. (1998) "ECOS: A framework for evolving connectionist and the ECOS learning paradigm", *ICONIP,* Kitakyushu, 232-1236.

Kasabov, N. (2001) "Evolving fuzzy neural networks for supervised/unsupervised on-line knowledge-based learning", *IEEE Transactions on Systems, Man, and Cybernetics,* **31**(6), 902-918.

Press, W.H., Flannery, B.P., Teukolsky, S.A. & Vetterling, W.T. (1987) *Numerical recipes in C*, Cambridge, Cambridge University Press.

Received: May. 10th 2004

Accepted in final format: Nov. 20th 2004

About the authors:

G. S. Ng received Bachelor of Mathematics with Major in Honours Computer Science and Minor in Combinatorics & Optimization and Electrical Engineering from University of Waterloo, Canada in 1983. He received Master of Applied Science in Electrical Engineering from the same university in 1985. He then further obtained his Ph.D. in Computer Engineering from the Nanyang Technological University, Singapore. Currently, he is an Associate Professor on the faculty of the Division of Computer Science in the School of Computer Engineering at Nanyang Technological University, Singapore. He also obtained

the Postgraduate Diploma of Teaching in Higher Education, from National Institute of Education, Singapore while he worked at the university. He has taught the course Computer Vision and Image Processing for 6 years, and Operating Systems for 14 years. Before joining Nanyang Technological University, he has worked as research staff at the National University of Singapore in the area of robotics vision for 2 years and has also worked in Singapore in the fields of automation engineering and software engineering for 3 years. During his industrial career, he has developed automatic visual inspection system for keyboard and hard disk head. He is a senior member of IEEE, and member of INS, ACM and PREMIER. Currently, he serves as a secretary of IEEE Singapore Computational Intelligence (formerly known as Neural Network) chapter as well as secretary of Pattern Recognition and Machine Intelligence Association (a member of International Association for Pattern Recognition). He has published 1 book, 2 book chapters and over 75 papers in international conferences and journals. He is listed in the Who's Who in the World 1996 and Who's Who in Science and Engineering 1998-1999. He is the Year-3 most popular lecturer recipient in 2004. His current interests are neural fuzzy systems with application in pattern recognition, image processing, biomedical and financial modeling. He can be reached by email asgsng@ntu.edu.sg

T Murali has completed Bachelor of Engineering (Computer Engineering) in Nanyang Technological University. He is currently working on a project involving Neural fuzzy systems and handwritten character classification. He can be reached by email asgsng@ntu.edu.sg

D. Shi is currently an assistant professor in School of computer engineering, Nanyang Technological University, Singapore. He got his BEng. in electrical engineering in 1991 and his first PhD in mechanical control in 1997, both from Harbin Institute of Technology, P. R. China. He obtained his second PhD in computer science in 2002 from University of Southampton, United Kingdom. He is a senior member of the IEEE. He can be reached by email asgsng@ntu.edu.sg

A. Wahab obtained his BSc. Honors degree in Telecommunication Engineering from Essex University, Colchester, England and his MSc. degree in Electronic Engineering from the National University of Singapore in 1979 and 1983 respectively. He then further obtained his Ph.D. in Computer Engineering from the Nanyang Technological University, Singapore. From 1979 to 1990, he worked for the Hewlett Packard Company in Singapore starting as a production engineer. In 1983, he became research and development project manager. While in HP, he worked in the field of input devices for personal computers and workstation. A. Wahab then joined the Nanyang Technological University of Singapore in the division of computing Systems as a senior lecturer in February 1990 and later as an associate professor. His main research interests are in the field of digital and optical computing and fuzzy neural networks. He is a member of the IEEE, ISCE and the SPIE. He can be reached by email asgsng@ntu.edu.sg