Neurocomputing 73 (2010) 613–621

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Use of the circle segments visualization technique for neural network feature selection and analysis

C.P. Lim *, S.L. Wang, K.S. Tan, J. Navarro, L.C. Jain

University of Science Malaysia, School of Electrical and Electronic Engineering, 14300 Nibong Tebal, Penang, Malaysia

ARTICLE INFO

Available online 26 November 2009

Keywords:
Circle segments
Neural network
Multi-layer perceptron
Data visualization

ABSTRACT

In this paper, the circle segments (CS) technique is proposed as a data visualization tool for selecting and analysing the effects of the input features towards the target outputs in constructing neural network models. Specifically, the multi-layer perceptron (MLP) network is employed to tackle function approximation and pattern classification tasks, and CS is used to provide visualization of the correlations between the input features and the target outputs in these tasks. The effectiveness of the proposed approach is evaluated using two benchmark data sets, one for function approximation and another for pattern classification. Performance comparison with the response surface methodology (in function approximation) and with principal component analysis (in pattern classification) is conducted. The results indicate the usefulness of CS in examining the correlations between the input–output data samples, with improved performances. In addition, a real medical diagnosis task is used to evaluate the applicability of the approach. The outcomes, again, demonstrate improvements in accuracy, sensitivity, and specificity with the use of CS for feature selection, even with more than 50% of the input features eliminated.

1. Introduction

Neural Networks (NNs) have been widely applied to tackle a lot of practical problems successfully owing to a number of salient properties. Perhaps one of the most important properties of NNs is the ability to learn from data samples, and to generalize the learned knowledge and to give predictions to previously new, unseen data samples. Usually, NNs are combined with other statistical data analysis methods to improve their performances. For example, NN and RSM (response surface methodology) were used to model a wire electrical discharge machining (WEDM) process in the manufacturing industry [1]. NN and RSM were also employed to build a predictive model for analysing the combined effects of process variables in extracellular protease production in the biochemical industry [2]. In the pharmaceutical industry, NN was coupled with the face-centred central composite design method to establish a model for the prediction of drug release profiles [3]. Another use of NN in the pharmaceutical industry was to access experimental data from a tablet compression [4].

While a NN has the capability of building a predictive model from a set of data samples, it usually acts like a black-box and provides only the prediction without any explanation. This approach is less convincing since the domain user cannot gain much information from the prediction. For data exploration and analysis to be effective and convincing, it is important to include the user in the process since humans have flexibility, creativity, and common sense in analysing data. Visualization techniques enable human's involvement in data exploration and analysis when the data samples are presented in a visual form. With the help of data visualization techniques, the user is allowed to access, identify, compare, verify, and understand a number of possible hypotheses associate with the data samples. As reported in studies [5–8], data visualization techniques help users to identify the significant input features, and derive useful information from the data set. Johansson et al. [5] used the SOM (self-organizing map) network combined with the parallel-coordinates method to analyse a set of molecular data. By tightly coupling these two techniques, the concept of visualized representations, instead of data points, made it easy to distinguish the overall structure of the underlying process model. An integration of a machine learning technique and data visualization was applied to extract clinically useful knowledge from a heterogeneous assortment of molecular data [6]. It was shown that Radviz (Radial visualization) helped one to understand important attributes from a large data set. Patrick et al. [7] also used Radviz to distinguish coding DNA sequences (exons) from non-coding DNA sequences (introns). Then, a rule-based NN was adopted to carry out data classification. Ruthkowska [8] demonstrated the use of a

* Corresponding author. Tel.: +60 4 5996033; fax: +60 4 5941023.
E-mail address: cplim@eng.usm.my (C.P. Lim).

0925-2312/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2009.06.018

visualization technique and the rule-based neural network in iris classification. The rule-based neural network was constructed by analysing the visualized representation of the data set.

From the above examples, it is clear that data visualization is useful in analysing the relationships of input features, and helping reduce redundancy and complexity in the data set before feeding the data samples to construct a predictive model. The main advantage of data visualization is that the process of data visualization normally is intuitive, and requires no prior understanding of complicated mathematical and statistical concepts of the user [9]. The user can interpret the possible relationship in the data samples based on his/her experience and perception, and obtain an overall picture of information hidden in the data samples. Data visualization can also be used to deal with missing features in a data set in an easy way, e.g., by allocating certain characteristics, i.e., colour, to represent the missing features.

A NN system generally focuses on its accuracy, instead of comprehensibility, in function approximation and pattern classification problems. Comprehensibility refers to how easily a system can be assessed by humans [10]. In this paper, the proposed method focuses not only on good performance of a NN model, but also comprehensibility of the data features. In essence, our approach uses a data visualization technique to analyse the effects of the input features with respect to the target outputs, and to select the significant features in building NN models. Feature selection is regarded as a search, among all possible transformations, for the best subspace that preserves class separation as much as possible in the lowest possible dimension space [11]. According to Huang and Wang [12], feature selection affects NN performance in a number of aspects, such as accuracy of the network, time needed for network training, number of data samples used learning and the cost associated with input features. Indeed, feature selection is important in NN training because the performance of the trained NN is directly related to the number of input features used. Irrelevant or uncorrelated input features can deteriorate the generalization of a NN model (as well as other data-based learning systems), owing to the "curse of dimensionality" problem [13,14]. The "curse of dimensionality" states that beyond a certain point, adding new input features could actually cause a reduction in the system performance. With fewer input features, a NN model has fewer weights to be adjusted, leading to better generalization and faster training [13]. Furthermore, feature selection enables the compression and use of pattern representation that is less sensitive to noise [11]. The importance of feature selection has led to the use of various techniques, such as principal component analysis (PCA) and genetic algorithm (GA), in NN and other machine learning systems in tackling a variety of problems [12–15].

In this paper, the circle segments (CS) visualization technique [9,16] is explored as a data visualization tool for feature selection and analysis in constructing NN models. Since our primary main aim is to explore the effectiveness of CS as a data visualization tool, we employ the most widely used NN architecture, i.e., the multilayer perceptron, [17] in this work. The proposed method, i.e., CS coupled with MLP, is applied to three case studies, i.e., two benchmark problems and one real problem that are related to function approximation and pattern classification. The results obtained are analysed, discussed, and compared.

The organization of this paper is as follows. Section 2 presents a description on the approach proposed used in this work, i.e., CS and MLP. Section 3 describes the case studies used in evaluating the effectiveness of the proposed approach. The results are presented, analysed, and discussed. Concluding remarks and suggestions for further work are presented in Section 4.

2. The proposed approach

2.1. Circle segments

The CS technique was proposed by Ankerst et al. [16]. The fundamental idea is to display data dimensions as segments of a circle. It uses one coloured pixel per data value. In [16], CS was used to display 10 years of stock data from the Frankfurt stock exchange. The data samples were arranged in such a way that the oldest data samples were at the middle of the circle, and the most recent samples were at the perimetre of the circle. However, in this paper, CS is used to identify the correlations between the input features and the output classes for constructing NN models. By visualizing the patterns in CS, analysis of the effects of the input features towards the output classes can be carried out, and insignificant input features can be identified and removed.

In general, CS comprises three main stages, i.e., dividing, ordering, and colouring. In the dividing stage, the circle is divided equally according to the number of input features. For example, assume that one is interested in a process that consists of one output and seven input features. The circle is divided into eight equal segments, with one segment representing the output, while others representing the inputs.

In the ordering stage, a proper way of sorting the data samples is needed to ensure that all input features are placed appropriately in the circle. Since the main aim in feature selection and analysis is to visualize the effects of each input feature with respect to the output class, the correlations between the input features and the output classes is used to sort the data samples accordingly (explained further in Section 2.1.1).

An example is presented to better illustrate the ordering stage. Assume that we have n pairs of data samples, $(\boldsymbol{x}_i, y_i), i = 1, \ldots, n$. Each data sample consists of a seven-dimensional input features, x_1, x_2,…, x_7, and a one-dimensional output class, y. The original input–output data samples are first normalized between 0.0 and 1.0. The combinations of the normalized input–output data samples are represented by a matrix, A, as in Fig. 1(a).

The correlations of the input features, (x_1, x_2,…, x_7), towards the output is denoted as r_{x1}, r_{x2},…, r_{x7}. Assume that the magnitudes of correlation are as described in Eq. (1).

$$R : r_{x2} > r_{x3} > \ldots > r_{x1} \tag{1}$$

Then, matrix A is first sorted based on the output column. When the output values are equal, the rows in A are further sorted based on the order specified in Q, which contains the ranked magnitudes of correlation in an ascending order, as shown in Eq. (2).

$$Q = [C_{x2}, C_{x3}, \ldots C_{x1}] \tag{2}$$

where C refers to the column order for each input feature. Based on this example, the rows in matrix A are first sorted by the output. When there is a tie in the output values, the rows in A are further sorted by the column of input x_2. When the elements in x_2 are equal, the rows in A are further sorted by the column of input x_3. This process continues until all data samples are sorted according to the column order specified in Q. After the ordering stage, assume that the newly sorted matrix A is shown in Fig. 1(b). The data samples in Fig. 1(b) are mapped into the circle-segments in such a way that data samples in the first row are placed at the centre of the circle, while data samples in the last row are placed at the perimetre of the circle, as shown in Fig. 2. The remaining data samples are placed accordingly into the circle-segments based on their row order.

In the colouring stage, colour is used to encode the relevance of each data value to the original one based on a colour map. This relevance measure is normalized in the range of 0–1. Fig. 2 shows an example of CS based on Fig. 1 and the associated colour map.

a

Data Sample	Output, y	Input, x_1	Input, x_2	...	Input, x_7
Sample 1	y_1	x_{11}	x_{12}	...	x_{17}
Sample 2	y_2	x_{21}	x_{22}	...	x_{27}
⋮	⋮	⋮	⋮	⋮	⋮
Sample n	y_n	x_{n1}	x_{n2}	...	x_{n7}

The matrix of input-output data is sorted based on the order of the columns, which are determined by their correlation magnitude.

b

Row Order	Output, y	Input, x_1	Input, x_2	...	Input, x_7
Order 1	y_5	x_{51}	x_{52}	...	x_{57}
Order 2	y_3	x_{31}	x_{32}	...	x_{37}
⋮	⋮	⋮	⋮	⋮	⋮
Order n	y_2	x_{21}	x_{22}	...	x_{27}

Fig. 1. An example of the ordering stage of the CS technique.

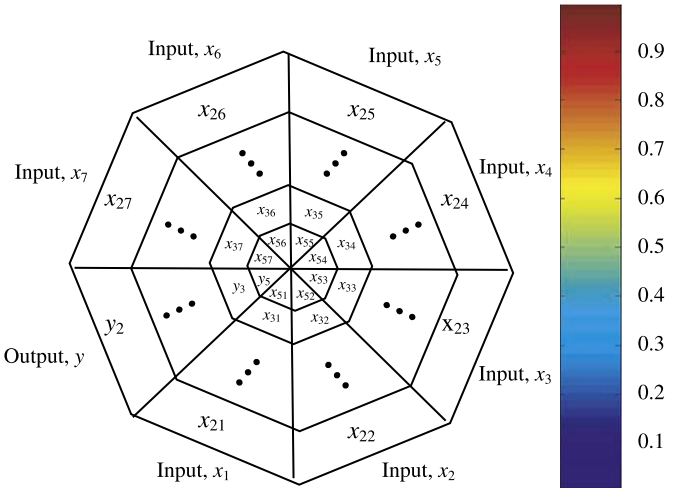

Fig. 2. The circle segments of the example in Fig. 1.

Based on the colour map located at the right side of Fig. 2, the highest and lowest values are represented by dark red and dark blue, respectively. The data samples within the y segment are transformed into colour values. The maximum value of y is represented by dark red, or has colour value=1. The minimum value is represented by dark blue, or has colour value=0. The values within the range are mapped linearly. By using the same procedure, the remaining input features within each segment are mapped according to the colour map. Therefore, a combination of colours along the perimetre represents a combination of the input–output. The combination of circle-segments with colouring according to relevance measure helps identify the patterns that are established between the input–output data.

2.1.1. Correlation

Correlation is used as reference to sort the data samples accordingly in the ordering stage. It is used to measure the relationship between the inputs and the outputs. The measure of correlation between the inputs and the outputs is invariant to the changes of scales. The correlation coefficient of a set of n input–output data pairs is computed using Eq. (3).

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}} \tag{3}$$

where $\bar{x}$ and $\bar{y}$ are the mean values of $(x_i, y_i), i=1,\ldots,n$, respectively. The correlation coefficient lies between -1 and $+1$. It gives a measure of the linear relationship between x and y. It is positive if high values of x are associated with high values of y. It is negative if high values of x are associated with low values of y. The magnitude of correlation as given in Eq. (3) is used as an indicator to sort the rows in matrix A.

2.1.2. Colour

Many of the graphic functions produce graphs that use colour as another data dimension. A full representation of colour displayed in digital storage systems is usually based on a three-component scheme. The commonly used colour space is the RGB (red, green, and blue) space. However, a large amount of memory is required for the storage of these three components. Therefore, the application of a limited palette of colours in computer systems has gained popularity [18]. This approach allows a small subset of all possible combinations of intensities of red, green, and blue to appear in a typical digital image based on a look-up table that stores the true colour. The representation uses a colour palette, or known as a colour map. A colour map is an m-by-3 matrix of real number between 0.0 and 1.0 with each row in the matrix defines an RGB vector that represents one colour. The m value refers to the length or size of the colour map.

In this paper, we use the pseudo-colour approach in mapping the values in each column in Fig. 1(b) to colours. In this approach, the minimum and maximum elements in each column are assigned to be the first colour and the last colour in the colour map. The values to be mapped to the first colour and the last colour in the colour map are denoted as *cmin* and *cmax*, respectively. The data values in between these values are linearly transformed from the second to the second last colours, using expression.

$$colourmap\_index = fix\left(\frac{(CData - c\min)}{(c\max - c\min)} \times cm\_length\right) + 1 \tag{4}$$

where *CData* refers to the input–output data, *cm_length* is the length of the colour map, and *fix*(*c*) is a function that rounds *c* toward zero [19]. The value of *colourmap_index* determines the type of colour to be used in the colour map. For example, if *colourmap_index*=*k*, the *k*th row of the colour map defines the *k*th colour according to the intensity of red, green, and blue in that row.

2.2. The multi-layer perceptron (MLP) network

As shown in Fig. 3, the MLP network consists of three main layers, i.e., input layer, hidden layer, and output layer. The hidden and output layers contain nodes which receives signals flowing from nodes in the previous layer, whereas the input layer contains nodes that receive the input features directly. The signals are scaled by parameters called weights and biases. The weights and biases are adjusted using a set of training data. The nodes sum all the incoming signals and produce an output through a transfer function of the sum. The node's outputs either are sent to other nodes in the following layer or become the system outputs. In the hidden and output layers, the net input to node *j* is represented by

$$I_j = \sum_{i=1}^{n} w_{ij}x_i + \theta_j \tag{5}$$

where x_i is the input, w_{ij} is the weight associated with each node connection, and θ_j is the bias associated with node *j*. This sum is sent through a transfer function $f(\cdot)$. Thus, the output of the node is

$$O_j = f(I_j) \tag{6}$$

Generally, the transfer functions used in MLP include log-sigmoid function, tan-sigmoid function, and linear function [20].

In MLP, the number of input and output nodes is uniquely determined by the number of input features and output classes. The important issue is how to appropriately set the number of hidden nodes. There are two extreme cases, either the network has too many hidden nodes, or it has too few. Too many hidden nodes can deteriorate the prediction capabilities of the network, while too few hidden nodes can obstruct the learning process. There are no specific guidelines to determine the optimum number of hidden nodes, except based one's experience. It is generally understood only that setting too few or too many hidden nodes causes lack-of-fit or over-fitting in the network. The trial-and-error method is normally used to set the network parameters, even though it is time consuming [21].

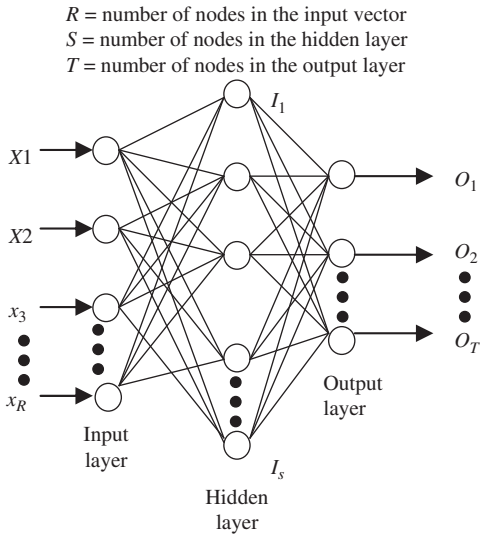

Fig. 3. The multi-layer perceptron network.

The training strategy in MLP is based on error-correction learning. During training, the network is fed with the input features along with their output classes, i.e., supervised learning. MLP learns the information that passes through the layers of nodes and predicts the outputs. The error in prediction is then fed backwards through the network to adjust the weights and bias in order to minimize the error. Usually, a data set is divided into three subsets, i.e., training set, validation set, and test set. The training set is used for network learning. The validation set is used for monitoring the generalization ability, and for determining when training should be stopped. The test set is used for evaluating the network performance.

3. Case studies

3.1. Benchmark problems

Two benchmark problems were first studied with different objectives. The first problem was related to modelling and prediction of the wire electrical discharge machining (WEDM) process as described in [1]. The objective was to demonstrate the ability of CS in analysing the effects of the input features towards the target output in function approximation problems. The RSM method was first used to construct a model for the WEDM process, and the CS diagram was used to provide visualization and insight of the input features towards the target output. MLP was used for prediction of the output, and the results obtained between RSM and MLP were compared.

The second problem was classification of the Wine data set obtained from the UCI machine learning repository [22]. The objective was to demonstrate how CS could be used to identify significant input features in pattern classification, and how the identified input features could lead to improvement in the performance of the MLP network. The results from MLP without feature selection (denoted as MLP), MLP with CS for feature selection (denoted as CS-MLP), and MLP with principal component analysis for feature selection (denoted as PCA-MLP) were analysed and discussed.

3.1.1. Experiment I

In this experiment, the objective was to demonstrate the use of CS for input feature analysis in NN-based function approximation problems. In this WEDM process, a slowly moving wire travels along a prescribed path and cut the work piece, with the discharge sparks acting like cutting teeth [1]. The problem was to construct a model relating four input features to one output target. The input features were the pulse-width (*A*), the time between two pulses (*B*), the wire mechanical tension (*C*), and the wire feed speed (*D*), and the target output was the surface roughness, R_a. The principles of design of experiments (DOE) were applied in collecting the relevant data samples. The DOE method allowed minimization of variability in the data samples arising from unwanted factors (e.g. noise) in an experiment.

The MINITAB software [23] was employed to carry out the RSM analysis. The analysis variance (ANOVA) was used to investigate the impact of the four input features towards the output. Table 1 presents the regression coefficients, the *t*-test results, and the associated *p*-values. A significance level, α, at 0.05 was chosen, i.e., any input feature with a *p*-value smaller than α was considered significant. As presented in Table 1, all four input features were significant as their *p*-values were smaller than 0.05. The interaction between pulse-width and wire feed speed (*A*∗*D*), as well as time between two pulses and wire feed speed (*B*∗*D*) were

Table 1
The estimated regression coefficients for R_a.

Term	Coefficient	*T*-test	*P*-value
Constant	3.2791	106.299	0.000
A	0.1571	9.432	0.000
B	−0.1126	−6.758	0.000
C	0.0552	3.313	0.004
D	−0.0496	−2.976	0.009
*A***D*	0.0543	2.661	0.017
*B***D*	0.0462	2.264	0.038

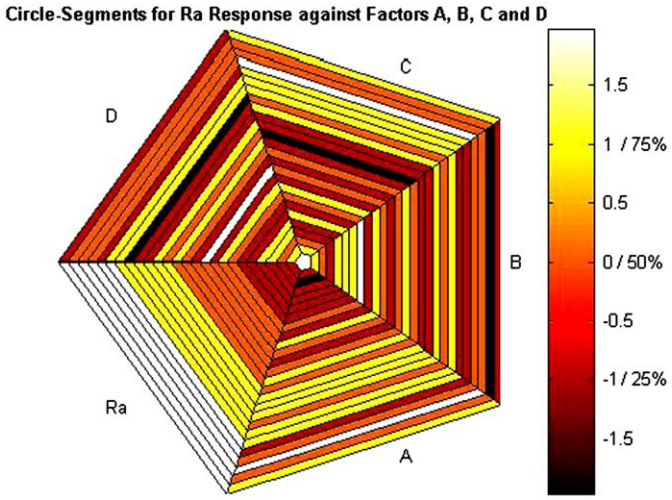

Fig. 4. The circle segments diagram of the WEDM problem.

also significant. Therefore, the target output, R_a, could be modelled using Eq. (7).

$$\hat{R_a} = 3.2791 + 0.1571A - 0.1126B + 0.0552C - 0.0469D + 0.0543AD + 0.0462BD \quad (7)$$

Based on the model in Eq. (7), a data set comprising the input–output samples was generated. The CS visualization technique was adopted to provide further insights into the correlations between the input features and the target output. As the WEDM process involved five variables (four input features and one target output), a CS diagram with five segments was constructed, as shown in Fig. 4. One of the segments represented the target output (R_a) while the remaining segments represented the four input features (denoted as *A*, *B*, *C* and *D*). The R_a segment was divided into four percentiles, and each percentile was represented by a colour. With the help of the colour map, the output values distributed within 25%, 50%, 75%, and above 75% were represented by red magenta, yellow, and white, respectively. The data samples in each input feature segment were mapped according to the colour map too. For example, in segment x_1, the maximum and minimum values were in white and black, respectively. The same procedure was repeated for the remaining factors. As the output was arranged from the minimum at the centre of the circle to the maximum at the perimetre of the circle, one could see changes of the colour range for each input features with respect to the output.

Based on Fig. 4, it could be observed that for segment *A* (the pulse-width), the region near the centre of the circle had low colour values, while the region near the perimetre of the circle had high colour values. This implied that an increase in *A* would lead to an increase in R_a. For segment *C*, a similar observation as in segment *A* could be obtained. By focusing on the centre of the circle, one could observe that segments *B* and *D* had high colour values. At the perimetre of the circle, both segments *B* and *D* had low colour values. In summary, the CS diagram revealed that *A* and *C* were directly proportional to R_a, while *B* and *D* were inversely proportional to R_a.

To confirm the analysis, the main effects plot, as shown in Fig. 5, was generated to examine the means of the various levels of each input features was compared with the overall mean. As can be observed in Fig. 5, an adjustment of *A* and *C* from a low level to a high level caused an increase in R_a. As for *B* and *D*, changes from a low level to a high level caused the opposite. Therefore, the analysis obtained from the CS diagram was in good agreement with that of the main effects plot. In other words, the CS diagram provided a good overview of the whole data set so that the domain user could analyse the effects of the input features toward the output, and gain further insight into the relations between the input–output data used in NN-based models in tackling function approximation problems.

The data samples were then used for MLP training. The performances of RSM and MLP were evaluated using the R^2 value using a separate test set. R^2 prediction provides an indication of the predictive capability of the model formed, and is commonly used in association with design of experiment (DOE) techniques including RSM. The MATLAB Neural Network Toolbox [20] was used to construct the MLP network. The scaled conjugate gradient (SCG) backpropagation [24] learning algorithm was adopted. SCG is a fast supervised learning method that aims to eliminate some of the disadvantages of the standard backpropagation learning algorithm of MLP, and it does not require the settings of learning rate and momentum constant [24]. Other network-related parameters were set according to the default values in the MATLAB Neural Network Toolbox [20]. The trial-and-error method was used to find the best combination of the numbers of hidden nodes and epochs, as follows.

The data samples were divided into three subsets: 31 samples for training, 10 samples for validation, and 7 samples for test. The number of hidden nodes was varied from 1 to 50. For each setting of the number of hidden nodes, the network was trained for 20,000 epochs (with an increment of 100). From the results, the numbers of hidden nodes and epochs that produced the best result was recorded. It was found that 13 hidden nodes with 4500 training epochs yielded the best performance.

Table 2 presents the performance comparison between RSM and MLP. From the R^2 prediction value, both models were satisfactory, as they could explain about 75.05% and 86.80% of the variability, respectively. Nevertheless, MLP achieved an improvement of 11.75% over RSM in R^2 prediction, thus indicating its usefulness in modelling the WEDM process.

3.1.2. Experiment II

In this case study, the CS diagram was deployed in a different way as compared with that in Experiment I. The objective was to examine the use of CS for input feature selection in NN-based pattern classification problems. The Wine data set with 13 input features (denoted by V1–V13) and three output classes was used for experimentation. There were 59, 71, and 48 samples in Class 1, Class 2, and Class 3 of the Wine data set, respectively.

Fig. 6 shows the CS diagram of the input–output data for Wine classification. The three output classes were represented by dark blue (Class 1), green (Class 2), and dark red (Class 3), respectively. Notice that segments V1, V6, V7, and V10 show significant colour changes for the three output classes. Among these four segments; each segment had different colour ranges that could segregate the classes, as summarized in Table 3. For segment V1, the colour

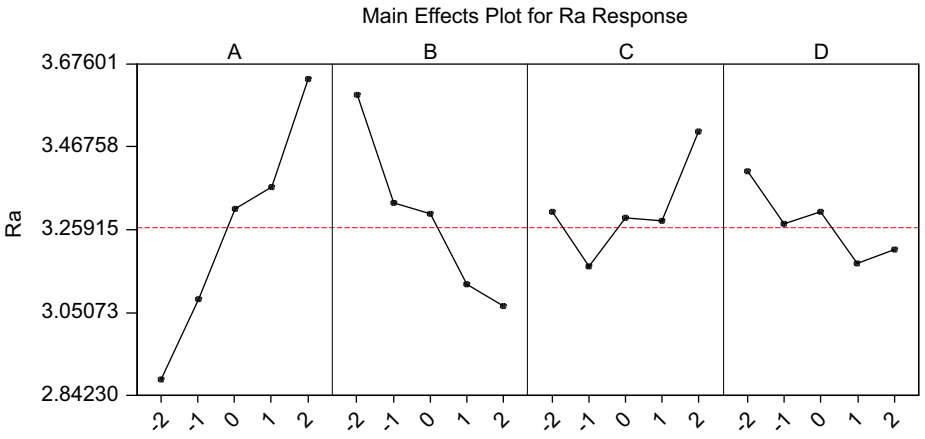

Fig. 5. The main effects plot of R_a.

Table 2
Performance comparison between RSM and MLP.

	Model		Improvement (%)
	RSM	MLP	
$R^2_{prediction}$ (%)	75.05	86.80	11.75

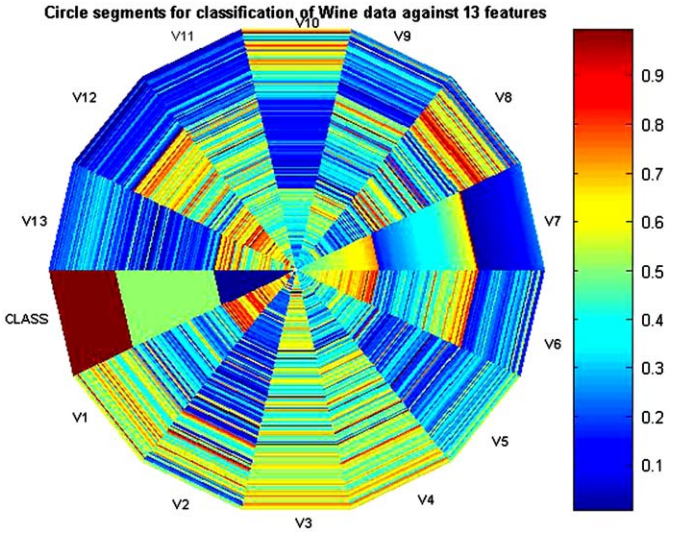

Fig. 6. The circle segments diagram for the Wine data set.

Table 3
Color ranges for V1, V6, V7, and V10.

Class	Colour range			
	V1	V6	V7	V10
Class 1	0.6∼1.0	>0.5	>0.5	0.3∼0.5
Class 2	0.0∼0.4	0.0∼0.4 (mostly)	0.0∼0.4 (mostly)	0.0∼0.3
Class 3	0.4∼0.8	0.0∼0.3	0.0∼0.2	>0.4

ranges for Class 1, Class 2, and Class 3 were 0.6 to 1.0, 0.0 to 0.4, and 0.4 to 0.8, respectively. Both segments V6 and V7 shows similar colour changes towards the output classes. For both segments V6 and V7, the colour ranges for Class 1 distributed above 0.5. For Class 2, most of the samples in Class 2 had colour ranges between 0.0 and 0.4, while the remaining samples had colour ranges above 0.4. As for Class 3, it had colour ranges below 0.3. In segment V10, the colour ranges for Class 1, Class 2, and Class 3 were 0.3 to 0.5, 0 to 0.3, and above 0.4, respectively.

For the remaining segments, colour overlapping was observed. As such, they were not effective in discriminating toward the output classes. For example, in segment V2, colour overlapping occurred within Classes 1 and 2, since both classes had colour ranges of 0.0–0.3. Based on the results obtained, V1, V6, V7, and V10 were selected as the significant input features for Wine classification using MLP.

Table 4 presents the eigenvalues and the associated variations of the principal components for the Wine data set. The first four principal components were selected as they accounted for 75.7% of the total variation. Table 5 presents the eigenvectors of PC1–PC4. Based on the results in Table 5, V6, V7, and V12 showed strong relationships with PC1; V1 and V10 with PC2; V2, V3, and V4 with PC3, and V8 with PC4. Thus, the input features selected for Wine classification consisted of V1–V4, V6–V8, V10, and V12.

Based on the CS and PCA analyses, three data sets were produced; one that consisted of all input features (i.e., the original data set), while another two data sets consisted of the input features selected using CS and PCA. These data sets were used for classification using the MLP network. The training, validation, and test sets comprised 105, 35, and 38 samples, respectively. As in the previous experiment, SCG was used as the learning algorithm of MLP, and other network-related parameters were set according to the default values in the MATLAB Neural Network Toolbox [20]. The trial-and-error was used to determine the set the numbers of hidden nodes and epochs, as follows.

The number of hidden nodes was varied from 1 to 50. For each hidden node setting, the number of epochs was varied from 100 to 10,000, with an increment of 100. Instead of finding only one best combination of hidden nodes and epochs, the best result obtained from the variation of epochs for each hidden node setting was recorded. Thus, a total of 50 results were produced, and the results were averaged in order to provide a stable indication of the network performance. This procedure was repeated for MLP, PCA-MLP, and CS-MLP.

The averages and standard deviations are presented in Table 6. The accuracy rate of MLP before feature selection was 92.26%. The feature set selected using PCA did not show improvement in accuracy. This is in contrast to that of CS, where CS-MLP achieved the best accuracy rate of 93.47%. Notice that the standard deviation of CS-MLP is the lowest. This implied that CS-MLP produced less variation in its results, as compared with those of

Table 4
Principal components for the Wine data set.

Principal	Eigenvalue	Proportion	Cumulative
PC1	0.22013	0.408	0.408
PC2	0.10245	0.19	0.597
PC3	0.04625	0.086	0.683
PC4	0.04012	0.074	0.757
PC5	0.03006	0.056	0.813
PC6	0.02517	0.047	0.859
PC7	0.01978	0.037	0.896
PC8	0.013	0.024	0.92
PC9	0.01228	0.023	0.943
PC10	0.01216	0.023	0.965
PC11	0.00746	0.014	0.979
PC12	0.00688	0.013	0.992
PC13	0.0044	0.008	1

Table 5
Eigenvectors of PC1 to PC4 for the Wine data set.

	PC1	PC2	PC3	PC4
V1	0.133	−0.551	0.084	0.041
V2	−0.249	−0.227	−0.491	−0.487
V3	0.001	0.163	−0.403	0.242
V4	−0.178	0.080	−0.477	0.081
V5	0.089	−0.188	−0.007	−0.016
V6	0.395	−0.074	−0.253	0.052
V7	0.415	−0.001	−0.196	0.027
V8	−0.333	−0.010	−0.287	0.709
V9	0.253	−0.031	−0.229	−0.077
V10	−0.092	−0.520	0.033	0.027
V11	0.251	0.237	0.106	0.353
V12	0.474	0.215	0.298	−0.083
V13	0.287	−0.444	0.152	0.230

Table 6
Classification results of the Wine data set.

Method	Test set accuracy (%)	
	Average	Standard deviation
MLP	92.26	8.887
PCA-MLP	90.37	4.773
CS-MLP	93.47	4.581

MLP and PCA-MLP; thus indicating stability in the CS-MLP performance in using the selected input features for Wine classification.

3.2. Medical diagnosis problem

To further evaluate the applicability of CS-MLP, a medical diagnosis problem using real stroke patient records was conducted. A data set with 18 input features comprising demography, medical history, physical examination information were extracted from the patient records, and a binary classification task was studied. The problem was to predict the Rankin Scale (RS) [25] category of patients upon discharge, i.e., either Class 1 (RS between 2 and 6, i.e., from slight disability to dead) or Class 2 (RS between 0 and 1, i.e., no symptoms at all to no significant disability despite symptoms). A total number of 661 cases with 520 in Class 1 and 141 in Class 2 were used. Three performance indicators that are commonly used in medical diagnostic problems are evaluated, i.e.,

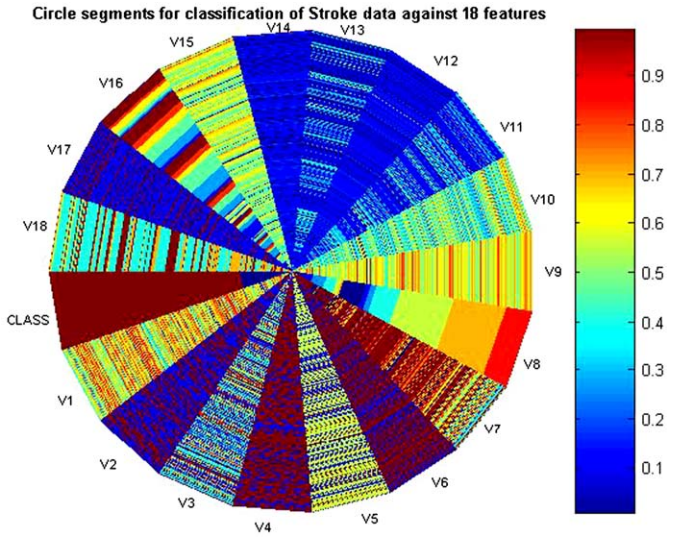

Fig. 7. The circle segments diagram for the Stroke data set.

Table 7
Principal components for the Stroke data set.

Principal	Eigenvalue	Proportion	Cumulative
PC1	0.33573	0.262	0.262
PC2	0.15896	0.124	0.386
PC3	0.14386	0.112	0.499
PC4	0.11533	0.09	0.589
PC5	0.09023	0.07	0.659
PC6	0.07917	0.062	0.721
PC7	0.07403	0.058	0.779
PC8	0.06239	0.049	0.828
PC9	0.0539	0.042	0.87
PC10	0.04268	0.033	0.903
PC11	0.03252	0.025	0.929
PC12	0.0308	0.024	0.953
PC13	0.02588	0.02	0.973
PC14	0.01546	0.012	0.985
PC15	0.00894	0.007	0.992
PC16	0.00584	0.005	0.996
PC17	0.00293	0.002	0.999
PC18	0.0016	0.001	1

Accuracy=Number of correctly classified cases/Total number of cases, Sensitivity=Number of correctly classified positives cases/Total number of positive cases, Specificity=Number of correctly classified negative cases/Total number of negative cases, (Positive and negative cases refer to Class 1 and Class 2, respectively)

Fig. 7 depicts the CS diagram with 18 input features (denoted as V1, V2, …, V18) and the output classes. Based on the CS diagram, one could observe that the data samples in this classification were dominated by Class 2. Therefore, the significant input features that could segregate Class 2 samples needed to be identified. Segments V8, V16, and V18 showed significant colour changes from the centre (Class 1) to the perimetre of the circle (Class 2). Based on segment V8, most Class 1 samples had colour range below 0.4. In this segment, most Class 2 samples were distributed within the colour range of 0.5–0.8. In V16, most Class 1 samples were distributed within colour range below 0.4, while Class 2 samples were dominated by colour range above 0.4. By observing V18, the centre of the circle had colour value of 0.7, while most Class 2 samples had colour value of around 0.4, with a few between 0.7 and 1.0. As such, V8, V16, and

Table 8
Eigenvectors of the PC1 to PC6 for the Stroke data set.

	PC1	PC2	PC3	PC4	PC5	PC6
V1	0.098	0.054	−0.083	0.042	−0.038	0.048
V2	0.611	−0.218	0.364	0.584	−0.316	0.024
V3	0.052	0.362	−0.064	−0.262	−0.741	−0.102
V4	0.520	−0.059	−0.188	−0.252	0.333	−0.410
V5	−0.054	−0.029	0.179	−0.244	−0.300	−0.023
V6	0.569	0.128	−0.027	−0.499	0.088	0.332
V7	0.039	−0.116	−0.389	0.090	−0.163	−0.683
V8	−0.035	0.122	0.407	−0.131	0.027	−0.170
V9	0.017	0.003	0.024	−0.049	−0.050	0.008
V10	−0.011	0.015	0.064	−0.073	−0.066	0.079
V11	0.006	−0.013	−0.023	0.045	0.114	0.136
V12	0.002	−0.011	−0.023	0.019	0.027	0.026
V13	0.018	−0.046	−0.054	0.059	0.082	0.142
V14	0.002	−0.015	−0.029	0.020	0.023	0.017
V15	−0.013	0.011	0.104	−0.072	−0.010	−0.071
V16	0.065	0.064	0.616	−0.217	0.148	−0.328
V17	0.086	0.870	−0.018	0.355	0.214	−0.083
V18	0.056	0.072	−0.267	−0.013	−0.136	0.216

Table 9
Input features that exhibit strong relationship with PC1 to PC6.

PC1	PC2	PC3	PC4	PC5	PC6
V2	V17	V8	V2	V3	V4
V4	–	V16	V6	–	V7
V6	–	–	–	–	–

Table 10
The overall results of the Stroke data set (the numbers in parentheses are the standard deviations).

Method	Accuracy (%)	Sensitivity (%)	Specificity (%)
MLP	83.18 (2.20)	91.47 (2.68)	51.79 (9.80)
PCA-MLP	84.01 (2.76)	93.23(2.46)	49.14 (11.12)
CS-MLP	86.72 (2.01)	93.43 (2.03)	61.29 (9.57)

V18 constituted the significant input features for this Stroke data set.

The cumulative values of PCA are presented in Table 7. It indicates that the first six principal components had accounted 72.1% of the total variation. Table 8 presents the eigenvectors of the six principal components, and Table 9 lists the variables that have strong relationship with each of the principal component. The input features in Table 9 exhibited eigenvectors that were larger in absolute value as compared with the others. Therefore, eight input features were selected for Stroke classification, i.e., V2–V4, V6–V8, and V16–V17.

Similar to the Wine problem, three data sets were produced and used to train the MLP network, i.e., the original data set, and the other two contained CS and PCA selected input features, respectively. The numbers of samples in the training, validation, and test sets were 395, 132, 134, respectively. The number of hidden nodes in the MLP was varied from 1 to 50. For each setting of hidden nodes, the number of epochs was varied from 100 to 10,000 (with an increment of 100). The best performance obtained from the variation of epochs for each hidden node setting was then recorded; thus producing 50 results. The same procedure was repeated for MLP, PCA-MLP, and CS-MLP.

The average results are summarized in Table 10. The results of CS-MLP were better than those of MLP and PCA-MLP. This happened for all the three performance indicators (accuracy, sensitivity, and specificity). The standard deviations of the CS-MLP results were relatively smaller than those from MLP and PCA-MLP too. This again indicated the stability in CS-MLP performances by using the selected input features.

3.3. Remarks

Based on the results from the benchmark and real problems, it is clear that use of CS as a data visualization tool provides a lot of benefits in feature selection and analysis for NN applications. The CS diagram is able to provide an overall picture pertaining to the correlations between the input features and the target outputs in a data set. In the WEDM prediction problem, the CS diagram is useful in analysing the effects of the input features toward the target output, and the analysis shows good agreement with that derived from using the main effect plot in statistical analysis. In the Wine and Stroke classification problems, the CS diagram helps differentiate between significant and insignificant input features. Eliminating the insignificant input features from the data sets helps reduce redundancy in the data sets; thus, leads to improved performances. In particular, all three performance indicators show improvement in the Stroke diagnosis task. In addition, the standard deviations reveal that the performance of the MLP with CS selected input features are stable, in comparison with those without feature selection or with PCA selected input features.

As most pattern classification problems involve a high-dimensional input space, using all the available input features can easily lead to "curse of dimensionality". The CS visualization technique for feature selection and analysis allows the user to understand the correlations between the input features and the target outputs in a data set. It enables the user to identify patterns and carry out necessary filtering of noisy input features, rather than feeding the whole data set for NN classification. In other words, CS allows the user to select only a subset of input features that are deemed significant and that provides the most discriminatory power in classification problems. With the implementation of CS as a features selection tool, an appropriate number of features which are representative can be determined while maintaining a high level of classification performance.

4. Summary

In this paper, CS, which is a data visualization technique, has been used for feature selection and analysis in NN applications. From the NN users' point of view, the CS technique is able to provide insights into the relationship of the whole input–output data in one glance. By identifying the patterns in the CS diagram, the correlation among input features can be observed, and the relevant and significant input features can be distinguished from the others. The use of CS with MLP allows the user to analyse the importance of each input feature towards the target output, thus eliminating those insignificant input features. From the experimental studies, it is observed that CS-MLP not only is able to improve the network results, but also exhibit stable network performance. The practical applicability of CS-MLP has also been demonstrated using a real medical diagnosis problem, whereby the accuracy, sensitivity, and specificity rates have been improved even with more than 50% of the input features eliminated.

In this paper, we have used CS as a useful data visualization tool for NN feature selection and analysis. However, more investigations are needed to further ascertain its effectiveness in different application domains. While the use of CS is intuitive and does not require strong mathematical and statistical knowledge, the process of interpreting the CS diagram is subjective and requires experienced users. Another difficulty is that the CS diagram becomes increasingly complex when more input features are used; leading to human fatigue. As a result, the CS diagram is

useful for analysing data with moderately high dimensions (perhaps fewer than twenty input features). One area for further work is to integrate CS, which is a qualitative technique, with other quantitative data analysis techniques such that the combined method could lead to comprehensibility and consistency in analysing high-dimensional data samples.

References

[1] T.A. Spedding, Z.Q. Wang, Study on modeling of wire EDM process, Journal of Materials Processing Technology 69 (1997) 18–28.
[2] J.R. Dutta, P.K. Dutta, R. Banerjee, Optimization of culture parameters for extracellular protease production from a newly isolated Pseudomonas sp. using response surface and artificial neural networks models, Process Biochemistry 39 (2004) 2193–2198.
[3] C.P. Lim, S.S. Quek, K.K. Peh, Prediction of drug release profiles using an intelligent learning system: an experimental study in transdermal iontophoresis, Journal of Pharmaceutical and Biomedical Analysis 31 (2003) 159–168.
[4] J. Bourquin, H. Schmidli, P. van Hoogevest, H. Leuenberger, Advantages of artificial neural networks as alternative modeling technique for data sets showing non-linear relationships using data from a galenical study on a solid dosage form, European Journal of Pharmaceutical Sciences 7 (1998) 5–16.
[5] J. Johansson, R. Treloar, M. Jern, Integration of unsupervised clustering, interaction and parallel coordinates for the exploration of large multivariate data, in: Proceedings of the Eighth International Conference on Information Visualization (IV'04), 2004 pp. 52–57.
[6] J.F. McCarthy, K.A. Marx, P.E. Hoffman, A.G. Gee, P. O'Neil, M.L. Ujwal, J. Hotchkiss, Applications of machine learning and high dimensional visualization in cancer detection, diagnosis, and management, Analysis New York Academy Science 1020 (2004) 239–262.
[7] P. Hoffman, G. Grinstein, K. Marx, I. Grosse, E. Stanley, DNA visual and analytic data mining, Proceedings on Visualization 97 (1997) 437–441.
[8] D. Ruthkowska, IF-THEN rules in neural networks for classification, in: Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'05), vol. 2, 2005, pp. 776–780.
[9] D.A. Kein, Information visualization and visual data mining, IEEE Transactions on Visualization and Computer Graphics 7 (1) (2002) 100–107.
[10] P. Meesad, G. Yen, Combined numerical and linguistic knowledge representation and its application to medical diagnosis, IEEE Transaction on Systems, Man and Cybernetics, Part A 33 (2003) 202–222.
[11] B. Lerner, M. Levinstein, B. Roserberg, H. Guterman, I. Dinstein, Y. Romem, Feature selection and chromosome classification using a multilayer perceptron, Proceedings of IEEE World Congress on Computational Intelligence 6 (1994) 3540–3545.
[12] C. Huang, C. Wang, A GA-based feature selection and parameters optimization for support vector machines, Expert Systems with Applications 31 (2) (2006) 231–240.
[13] J.C.B. Melo, G.D.C. Cavalcanti, K.S. Guimarães, PCA feature selection for protein structure prediction, Proceedings of the International Joint Conference on Neural Networks 4 (2003) 2952–2957.
[14] L.J. Cao, W.K. Chong, Feature extraction in support vector machine: a comparison of PCA, KPCA and ICA, in: Proceedings of the 9th International Conference on Neural Information Processing, vol. 2, 2002, pp. 1001–1005.
[15] A.K. Mishra, B. Mulgrew, Radar signal classification using PCA-based features, IEEE International Conference on Acoustics, Speech and Signal Processing 3 (2006) 1104–1107.
[16] M. Ankerst, D.A. Keim, H.P. Kriegel, Circle segments: a technique for visually exploring large multidimensional data sets, Proceedings of Visualization '96, Hot Topics Session, 1996.
[17] D.E. Rumehalt, G. Hinton, G. Williams, Learning internal representation by error propagation, second ed., In Parallel Distribution Processing, vol. 1, MIT Press, 1986.
[18] S.J. Sangwine, R.E.N. Horne, The Colour Image Processing Handbook, first ed, Chapman&Hall, London, 1998.
[19] Fixed-Point Toolbox User's Guide, The MathWorks, Inc, 2008.
[20] H. Demuth, M. Beale, M. Hagan, Neural Network Toolbox User's Guide, The Math Works, Inc., 2008.
[21] W. Sukthomya, J. Tannock, The optimization of neural network parameters using Taguchi's design of experiments approach: an application in manufacturing process modeling, Neural Computing and Applications 14 (2005) 337–344.
[22] A. Asuncion, D.J. Newman, UCI Machine Learning Repository 〈http://www.ics.uci.edu/~mlearn/MLRepository.html〉. Irvine, CA: University of California, School of Information and Computer Science, 2007.
[23] Meet Minitab 15, Minitab Inc., 2007.
[24] M.F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, Neural Networks 6 (1993) 525–533.
[25] P.W. New, R. Buchbinder, Critical appraisal and review of the rankin scale and its derivatives, Neuro epidemiology 26 (2005) 4–15.

Chee-Peng Lim received his BEng (Electrical) degree from University of Technology Malaysia in 1992, and both the M.S.c in Engineering (Control Systems) and Ph.D. degrees from University of Sheffield, UK, in 1993 and 1997. He is currently a professor at School of Electrical & Electronic Engineering, University of Science Malaysia. He has published more than 150 technical papers in books, international journals, and conference proceedings. His research interests include soft computing, pattern recognition, medical prognosis and diagnosis, fault detection and diagnosis, condition monitoring.

Shir Li Wang obtained her bachelor degree (1999 – 2002) and master degree (Nov 2004–Mar 2007) from the University of Science Malaysia, Malaysia. She is now a Ph.D. candidate of the Australian Defence Force Academy at the University of New South Wales (UNSW@ADFA), Australia. Her research interests focus on neural networks, clustering, ensemble and data mining.

Dr. Kay Sin Tan, FRCP(Edin), is a Associate Professor and Consultant Neurologist at the Faculty of Medicine, University of Malaya, Kuala Lumpur, Malaysia. He was trained and certified in internal medicine, neurology, and cerebrovascular medicine. His research interests include large clinical trials in stroke, molecular biology of young stroke patients, transcranial and extracranial Doppler ultrasound studies and medical applications of artificial intelligence systems. He was a Stroke Fellow and a Visiting Fulbright Scholar to the Wake Forest University Hospital, Winston-Salem, North Carolina, USA in 2002/2003 and Visiting Fellow to The National Stroke Institute, Melbourne, Australia in 2008. He has published over 20 original peer reviewed articles and several book chapters.

Dr. Jose Navarro is currently the Chairman of the Department of Neurology at the Jose R Reyes Memorial Medical Center and the head of the Intensive Care Unit and Acute Stroke Unit and Neurovascular Laboratory of the Philippine heart center. He is currently second Vice-President of the Stroke Society of the Philippines and a member of the board of Directors for Asia, International Stroke Society. Dr. Navarro is active in Stroke research in the Philippines and has participated in international stroke trials such as VITATOPS, CLAIR, ENOS and SAINT-2.

Lakhmi C. Jain is a Director/Founder of the Knowledge-based Intelligent Engineering Systems (KES) Centre, University of South Australia. He is a fellow of the Engineers Australia. He has initiated a postgraduate stream by research in the KES area. His interests focus on the applications of novel techniques such as knowledge-based systems, virtual systems, multi-agent intelligent systems, artificial neural networks, genetic algorithms, and the application of these techniques.