POP-TRAFFIC: A Novel Fuzzy Neural Approach to Road Traffic Analysis and Prediction

Chai Quek, Michel Pasquier, and Bernard Boon Seng Lim

***Abstract*—Although much research has been done over the decades on the formulation of statistical regression models for road traffic relationships, they have been largely unsuitable due to the complexity of traffic characteristics. Traffic engineers have resorted to alternative methods such as neural networks, but despite some promising results, the difficulties in their design and implementation remain unresolved. In addition, the opaqueness of trained networks prevents understanding the underlying models. Fuzzy neural networks, which combine the complementary capabilities of both neural networks and fuzzy logic, thus constitute a more promising technique for modeling traffic flow. This paper describes the application of a specific class of fuzzy neural network known as the pseudo outer-product fuzzy neural network using the truth-value-restriction method (POPFNN-TVR) for short-term traffic flow prediction. The obtained results highlight the capability of POPFNN-TVR in fuzzy knowledge extraction and generalization from input data as well its high degree of prediction capability as compared to traditional feedforward neural networks using backpropagation learning.**

***Index Terms*—Density and volume prediction, fuzzy neural network, noise tolerance, pseudo outer-product-based fuzzy neural network (POPFNN), traffic analysis, traffic prediction and modeling, truth-value-restriction fuzzy inference.**

I. Introduction

THE FIELD of transportation studies has in recent years seen an increased interest in neural network applications due to the inability of traditional methods to address the complexity of road traffic characteristics and relationships. Results that are comparable or better than that obtained using mathematical models—typically statistical regression models—have thus been achieved using neural networks in areas such as driver behavior prediction, road pavement maintenance, vehicle detection and classification, congestion detection, traffic control analysis, and intelligent transmission control [1]–[13]. Although some results are promising, the difficulties in the design and implementation of neural networks remain unresolved and the opaqueness of the trained networks prevents understanding the underlying models. A possible solution is to adopt instead a knowledge-based approach that consists of automatically generating a set of expert rules to model the problem and subsequently using the rules independently. In the case of traffic flow and driver behavior modeling, such rules can represent those characteristics that are invaluable to traffic engineers. The model thus obtained becomes useful in developing highway and transportation plans, performing economics analysis, establishing geometric design criteria, selecting and implementing traffic control measures, as well as evaluating the service quality of transport facilities. In addition, this model provides traffic engineers with the basis for a platform for transportation planning and simulation. Most notably are recent efforts in the application of fuzzy neural techniques to model traffic patterns [7], [11].

Fuzzy neural networks, which combine the capabilities of both neural networks and fuzzy logic, are seen as a very promising technique for automatically deriving from experimental data an approximate rule-based model. This paper describes a novel approach to the analysis and prediction of traffic flow using a specific class of fuzzy neural networks known as the pseudo outer-product fuzzy neural network using the truth-value-restriction method (POPFNN-TVR), which was developed at the Centre for Computational Intelligence, formerly Intelligent Systems Laboratory, Nanyang Technological University, Singapore [14]. As the original POPFNN-TVR implements a fixed-label architecture that is not suitable for traffic modeling, it was modified into a generic fuzzy neural system that can accommodate multiple labels. The system was then trained and employed specifically for short-term traffic flow prediction and interlane traffic modeling. The former is elaborated in the following sections, whereas the latter is described in a forthcoming paper. The results obtained highlight the capability of POPFNN-TVR to extract a working fuzzy knowledge rule base as well as generalize from the input data. Experimental results also demonstrate that POPFNN-TVR possesses a higher degree of prediction capability as well as better noise tolerance than feedforward neural networks using conventional backpropagation learning (FFBP) [15].

II. Traffic Flow Analysis

This section briefly describes fundamental traffic flow characteristics and associated analytical techniques, knowledge of which is essential in the planning, design, and operation of transportation systems.

A. Macroscopic and Microscopic Analysis

A transportation system can be viewed as consisting of many moving particles that interact with one another as well as with the environment. Traffic flow analysis can be performed using either microscopic parameters, which relate to the behavior of individual vehicles in the traffic stream with respect to each

Manuscript received November 6, 2004; revised July 13, 2005 and October 7, 2005. The Associate Editor for this paper was H. Dia.

C. Quek and M. Pasquier are with the Centre for Computational Intelligence (formerly Intelligent Systems Laboratory), Nanyang Technological University, Singapore 639798 (e-mail: ashcquek@ntu.edu.sg; asmbpasquier@ntu.edu.sg).

B. B. S. Lim is with Ernst & Young Laboratories, Singapore 049315, and also with the Institute of Systems Sciences, Singapore 119615.

Digital Object Identifier 10.1109/TITS.2006.874712

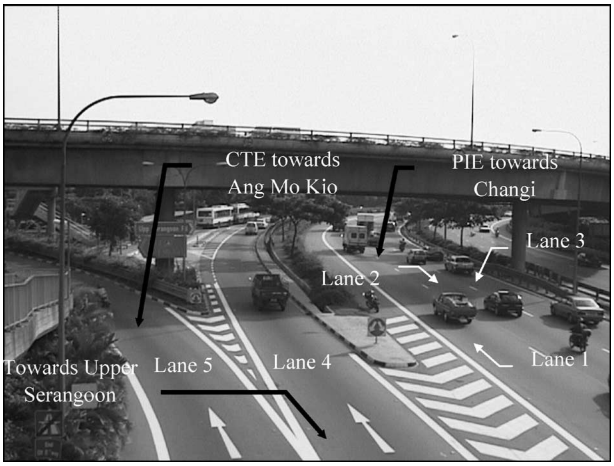


Fig. 1. Photograph of the Pan-Island Expressway site 29 in the direction toward Changi.

other, or macroscopic parameters, which describe the traffic stream as a whole. Microscopic parameters include characteristics such as spacing and headway, which refer to the distance between the front bumpers of successive vehicles in a traffic stream and the time delay between successive vehicles as they pass a particular road location, respectively. Both the spacing and headway of individual vehicles vary over a range of values that is generally based on the speed of the traffic stream. Therefore, in aggregate, microscopic parameters are related to the macroscopic flow parameters: speed, volume, and density.

On one hand, microscopic analysis may be selected for moderate-size systems, where the number of vehicles is relatively small, and there is a need to study their individual behavior. On the other hand, macroscopic analysis may be selected for higher density larger scale systems, for which a study of the global behavior of groups of vehicles is sufficient. Because the main objective of this project is to model a highway section as a whole, macroscopic analysis is a more appropriate choice. This work differs from [11] as it employs raw data as input as opposed to a transformed space (wavelet). Furthermore, the logical inference system in [14] is based on theoretical truth-value-restriction fuzzy inference paradigm [33].

B. Field Data Collection

The set of data employed in this project is described hereafter. Both vehicle classification counts and speed data were collected from a five-lane section along the Pan Island Expressway, Singapore, in both the eastward and westward directions toward Changi and Jurong, respectively. Site 30 is the upstream station located between the Toa Payoh South Fly-over and Thomson road, whereas site 29, which is shown in Fig. 1, is located further down the stream near Kim Keat Avenue. Data samples were collected from inductive and piezoelectric loop detectors installed beneath the road surface by the Land Transport Authority of Singapore in 1996 to facilitate traffic data collection, as depicted in Fig. 2. Detectors are connected to a traffic counter, which, in this case, is a microprocessor-based Marksman 660 traffic counter classifier.

The loop detectors are capable of recording vehicle count, vehicle speed, and vehicle classification of up to 13 categories. They are not perfect, however, and errors such as vehicle misclassification and undercounting are unavoidable. Video recordings were thus taken from overhead bridges next to the respective counting stations to monitor the actual traffic flow conditions during data collection. The data used in this project were collected at 5-min intervals over a period of 6 days, with the Marksman 660 configured to classify speed data and vehicle counts into nine speed and 13 vehicle categories, as shown in Tables I and II, respectively.

III. Fuzzy Neural Networks

Fuzzy neural networks are hybrid intelligent systems that possess the advantages of both neural networks and fuzzy systems—in the former, the learning and optimization abilities, as well as the connectionist structure; and in the latter, the human-like reasoning capacity and the ease of incorporating expert knowledge [16]. In addition, such hybrid intelligent systems alleviate the shortcomings of the respective techniques. These include common problems encountered in the design of fuzzy rule-based systems; for example, the determination of the membership functions, the identification of the fuzzy rules, as well as the fuzzy inference process. Such problems in fuzzy rule-based systems can be resolved in fuzzy neural networks using neural net techniques. One well-acknowledged drawback of neural networks is their opaqueness. The integration of fuzzy concepts in fuzzy neural systems greatly improves the transparency for a better understanding of their inner workings. Each membership function and fuzzy rule used in fuzzy neural networks have clear semantic meanings, hence, the neural-network-like structure of fuzzy neural networks becomes increasingly transparent once they are trained. In addition, the choice of the initial values for the parameters is made simpler in fuzzy neural networks.

An important aspect in the design of fuzzy neural networks is the identification of the fuzzy rules. However, there is no systematic design procedure at present. The recent research direction in the identification of the fuzzy rules in fuzzy neural networks is to learn and modify the rules from past experience. Currently, much research work that uses linguistic as well as numerical information to generate and adapt fuzzy rules has been reported [7], [14], [17]–[23], [29]–[31]. They can be categorized into the following approaches:

Approach 1: This approach uses linguistic information to identify fuzzy rules in the fuzzy neural networks prior to the application of neural network techniques to adjust the rules [18], [24]–[27]. The incorporation of linguistic information prevents the random choice of the initial fuzzy rules employed in the system. Consequently, the system converges faster during training and performs better. However, the approach is rather subjective because linguistic information from experts may vary from person to person, and from time to time.

Approach 2: This approach uses unsupervised learning algorithms to identify fuzzy rules in the fuzzy neural

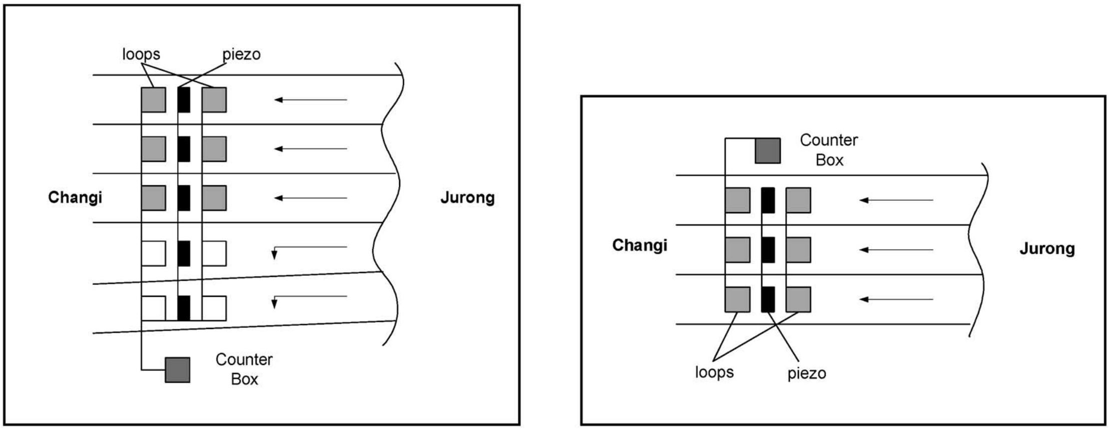


Fig. 2. Five-lane and three-lane road sections at sites 29 and 30 of the Pan-Island Expressway.

TABLE I
SUMMARY OF SPEED BIN

Speed Bins	Range (km/h)	Median (km)
1	0 ~ 40	20
2	40 ~ 50	45
3	50 ~ 60	55
4	60 ~ 70	65
5	70 ~ 80	75
6	80 ~ 90	85
7	90 ~ 100	95
8	100 ~ 110	105
9	110 ~ 120	115

networks prior to the application of neural network techniques to adjust the rules [17], [19], [28]–[31]. Because the training data set is the only source of information employed in the second approach, it must therefore be representative. Otherwise, the derived fuzzy rules will be ill defined and the resultant fuzzy neural network will not model the behavior inadequately. Moreover, the unsupervised learning algorithms that are used in the rule identification process must be carefully selected in the absence of experts' opinions.

Approach 3: This approach uses supervised learning algorithm (particularly the backpropagation technique) to identify the fuzzy rules in the fuzzy neural networks [7], [20], [21]. These fuzzy neural networks are essentially multilayered with the inputs and outputs as fuzzy membership values that satisfy certain constraints. The backpropagation learning algorithm [32] is often utilized in such fuzzy neural networks to produce the mapping from inputs to outputs. However, the fuzzy neural network appears as a "black box" at the end of the training process. The actual model acquired through training and the semantics of the structure remain opaque. The opaqueness contradicts the original intention for the integration of fuzzy concepts in neural networks.

IV. POPFNN-TVR FUZZY NEURAL NETWORK

This section describes the fuzzy neural network used in this project for traffic flow analysis and prediction.

A. Characteristics of Pseudo Outer-Product-Based Fuzzy Neural Network (POPFNN)

The POPFNN is a hybrid system that possesses the advantages of both neural networks and fuzzy systems. It belongs to a class of fuzzy systems that are constructed using numerical data with their parameters adjusted using neural network techniques. In our case, the initial set of parameters and the system structure are not derived from linguistic information provided by experts but constructed from a set of training data using an unsupervised learning algorithm and fine tuned on the basis of the numerical information [19]. The functions performed by each layer of the POPFNN correspond to the inference steps of the truth-value-restriction method [33].

One of the most important aspects of fuzzy neural networks is the identification of fuzzy rules, using unsupervised learning algorithms such as the self-organizing algorithm [34] and competitive learning algorithm [35]. The rule identification method used in POPFNN is the one-pass lazy pseudo outer-product (LazyPOP) learning algorithm [22], which is closely related to Hebbian's learning law. The use of POPFNN as the fuzzy neural network for traffic flow prediction has the advantages to be efficient, convenient, highly intuitive, and easier to understand than other rule identification algorithms.

B. Structure of POPFNN

The POPFNN [14] has a five-layer structure, as depicted in Fig. 3, where each layer is characterized by the fuzzy operation that it performs, namely input (fuzzification), condition, rule combination, consequence, and output (defuzzification). These operations are elaborated in the following subsections.

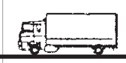
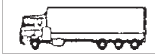
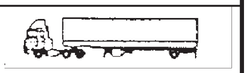
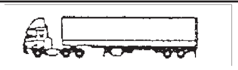
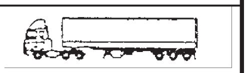
Input Layer of POPFNN: Neurons or linguistic nodes in the input layer represent input linguistic variables such as "height" and "width" and directly transmit nonfuzzy input values to the second layer. The net input f_i^{I} and net output o_i^{I} of the ith linguistic node L_i are given as follows:

$$f_i^{\mathrm{I}} = x_i \quad o_i^{\mathrm{I}} = f_i^{\mathrm{I}} \tag{1}$$

where x_i is the ith element of the input vector $\mathbf{X}$.

Condition Layer of POPFNN: Neurons or input-label nodes in the condition layer represent labels such as "small,"

TABLE II
VEHICLE CLASSIFICATION TABLE

Vehicle Classification Table		GR01-FHWA13
1	Motorcycle	
2	Car	
	Car and 1-Axle Trailer	
	Car and 2-Axle Trailer	
3	Light Goods Vehicle (LGV)	
	LGV and 1-Axle Trailer	
	LGV and 2-Axle Trailer	
	LGV and 3-Axle Trailer	
4	Bus 2-Axle	
	Bus 3-Axle	
5	Rigid 2-Axle Truck (HGV)	
6	Rigid 3-Axle Truck (HGV)	
7	Rigid 4 or more Axle Truck (HGV)	
	Rigid 4 or more Axle Truck (HGV)	
8	Artic, 2-Axle Tractor & 1-Axle Semi-Trailer	
	Artic, 2-Axle Tractor & 2-Axle Semi-Trailer	
	Artic, 3-Axle Tractor & 1-Axle Semi-Trailer	
9	Artic, 2-Axle Tractor & 3-Axle Semi-Trailer	
	Artic, 3-Axle Tractor & 2-Axle Semi-Trailer	
10	Artic, 2-Axle Tractor & 4-Axle Semi-Trailer	
	Artic, 3-Axle Tractor & 3-Axle Semi-Trailer	
11	Multi-Trailer Truck, ≤ 5-Axle	
12	Multi-Trailer Truck, ≤ 6-Axle	
13	Multi-Trailer Truck, ≥ 7-Axle	

"median," and "large" of the corresponding input linguistic variables and constitute the antecedents of the fuzzy rules. Each input-label node $\mathrm{IL}_{i,j}$ denotes the jth input label of the linguistic node L_i in the input layer and is modeled as a bell-shaped membership function to ensure differentiability, as required by the backpropagation algorithm [15] employed during the final phase of the learning process. The net input $f_{i,j}^{\mathrm{II}}$ and net output $o_{i,j}^{\mathrm{II}}$ of the input-label node $\mathrm{IL}_{i,j}$ are given as follows:

$$\text{net input}: f_{i,j}^{\mathrm{II}} = -\frac{\left(o_{i,j}^{\mathrm{I}} - c_{i,j}^{\mathrm{II}}\right)^2}{\delta_{i,j}^{\mathrm{II}}} \tag{2}$$

where $c_{i,j}^{\mathrm{II}}$ and $\delta_{i,j}^{\mathrm{II}}$ are the centroid and the width of the membership function for the input-label node $\mathrm{IL}_{i,j}$, respectively.

Rule-Base Layer of POPFNN: Neurons or rule nodes in this layer represent fuzzy rules such as "if height is short, then weight is light." The net input f_k^{III} and net output o_k^{III} of the kth rule node R_k are given as follows:

$$\begin{aligned} \text{net output}: o_{i,j}^{\mathrm{II}} &= e^{f_{i,j}^{\mathrm{II}}} \\ \text{net input}: f_k^{\mathrm{III}} &= \min_{i,j}\left(o_{i,j}^{\mathrm{II}}\right) \\ \text{net output}: o_k^{\mathrm{III}} &= f_k^{\mathrm{III}} \end{aligned} \tag{3}$$

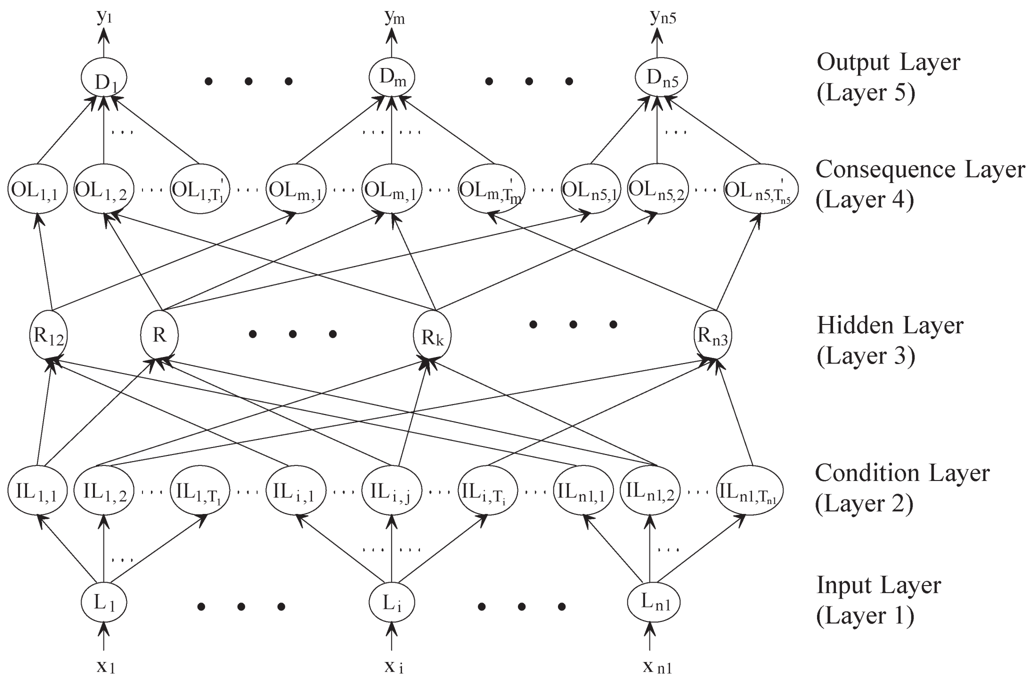


Fig. 3. Structure of the POPFNN.

where $o^{\rm II}_{i,j}$ is the net output of the input-label node $\mathrm{IL}_{i,j}$, which forms one antecedent condition to the rule node R_k.

Consequence Layer of POPFNN: Neurons or output-label nodes in the consequence layer represent labels such as "light," "median," and "heavy" of the corresponding output variables. Output-label node $\mathrm{OL}_{l,m}$ represents the lth label of the output node D_m. The net input $f^{\rm IV}_{l,m}$ and net output $o^{\rm IV}_{l,m}$ of the output-label node $\mathrm{OL}_{l,m}$ are given as follows:

$$\text{net input}: f^{\rm IV}_{l,m} = \sum_k o^{\rm III}_k$$

$$\text{net output}: o^{\rm IV}_{l,m} = \min\left(1, f^{\rm IV}_{l,m}\right) \tag{4}$$

where $o^{\rm III}_k$ is the net output of the rule node R_k. Note that (4) is applicable only for the rule nodes that take the output-label node $\mathrm{OL}_{l,m}$ as one of their consequences.

Output Layer of POPFNN: Neurons or defuzzification nodes in the output layer represent output variables such as "weight." The net input $f^{\rm V}_m$ and net output $o^{\rm V}_m$ of the mth defuzzification node D_m are given as follows:

$$\text{net input}: f^{\rm V}_m = \sum_{l=1}^{T'_m} \left(c^{\rm IV}_{l,m}/\delta^{\rm IV}_{l,m}\right) \times o^{\rm IV}_{l,m}$$

$$\text{net output}: o^{\rm V}_m = \frac{f^{\rm V}_m}{\sum\limits_{l=1}^{T'_m} \left(o^{\rm IV}_{l,m}/\delta^{\rm IV}_{l,m}\right)} \tag{5}$$

where $c^{\rm IV}_{l,m}$ is the centroid of the membership function for the output-label node $\mathrm{OL}_{l,m}$, and $\delta^{\rm IV}_{l,m}$ is the width of the membership function for the output-label node $\mathrm{OL}_{l,m}$.

V. Learning Process of POPFNN

The learning process of POPFNN consists of three phases, namely 1) self-organization, 2) POP learning, and 3) supervised learning. Similar to many works on fuzzy neural networks, e.g. [36], a self-organizing algorithm is first employed to initialize the membership functions of both input and output variables by determining their centroids and widths [34]. Next, the LazyPOP learning algorithm is used to identify the fuzzy rules that are supported by the training data. Lastly, the derived structure and parameters are fine tuned using the backpropagation learning algorithm [15].

Most existing fuzzy neural networks make use of self-organizing algorithm [18] to identify fuzzy rules. The rationale behind such approaches is that if there exists a rule that maps an input space X to an output space Y, then there will exist a corresponding cluster of representative data samples in the joint input–output space. As a result, the input and output spaces are frequently regarded as a joint space, and the data pairs in this joint space are treated like augmented data. That is, given a group of data samples described as follows:

$$\left(x^{(1)}, y^{(1)}\right), \left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(r)}, y^{(r)}\right), \ldots, \left(x^{(p)}, y^{(p)}\right)$$
$$\text{for } x^{(r)} \in D1 \text{ and } y^{(r)} \in D2$$

which is associated with the rule "if x is A, then y is B," we can rewrite these ordered pairs into an augmented data set described as follows:

$$\left\{d^{(1)}, d^{(2)}, \ldots, d^{(r)}, \ldots, d^{(p)}\right\}$$
$$d^{(r)} = \left(x^{(r)}, y^{(r)}\right) \in D1 \times D2.$$

This augmented data set is then used to discern clusters in the joint space $D1 \times D2$. Each cluster in the joint space

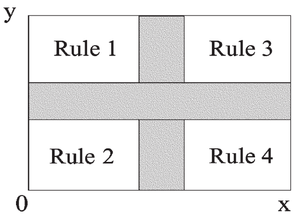


Fig. 4. Clusters in input–output as rules.

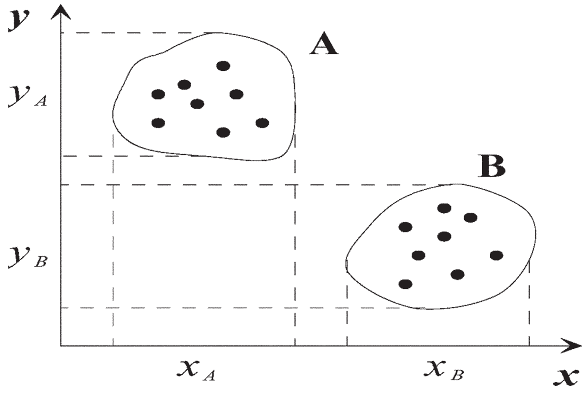


Fig. 5. Clusters in input–output space and their projections.

$D1 \times D2$ corresponds to a rule that maps the input space $D1$ to the output space $D2$ as shown in Fig. 4. The checkered area in Fig. 4 represents the fuzzy boundary between clusters. This clustering process is often accomplished by using a self-organizing algorithm [18].

However, in most integrated fuzzy neural networks, where the membership functions are determined prior to the identification of the fuzzy rules, self-organizing or competitive learning becomes redundant. In such cases, the boundary of the clusters in the input and output spaces have already been predefined. Thus, it is unnecessary to use self-organizing algorithms to determine the fuzzy partitions of the input–output space.

Clusters or subsets of data samples in the input and output spaces map onto clusters in the joint space (see Fig. 5). As a result, once the clusters in the input and output spaces are known, the clusters in their joint space are therefore apparent to some degree. Some efforts have been attempted to make use of the known membership functions to find the fuzzy rules. An example is Lin and Lee's work [17], where competitive learning is employed to identify the fuzzy rules. In his work, all the possible fuzzy rules must be listed before the initiation of competitive learning. These fuzzy rules are then repeatedly tested against the training data to find their most likely consequence. After the competitive learning, the link with the largest weight is selected, and the consequence that it connects to is subsequently held as the real consequence of the rule. Although Lin and Lee's method does integrate information of clusters in the input and output spaces to some degree, it involves iterative training before the system comes to a stable state. To more effectively utilize the information derived from the identified membership functions, we proposed a new rule-identification algorithm in this paper, called the pseudo outer-product (POP) learning algorithm. The proposed POP learning algorithm is a one-pass learning algorithm, and in this sense, it is superior to Lin's competitive learning, which is iterative. To correctly identify all the relevant fuzzy rules, it is only necessary to feed the training data through POPFNN once using the POP learning algorithm. Furthermore, the POP learning algorithm is easy to comprehend because it coincides with our intuitive ways to identify relevant rules.

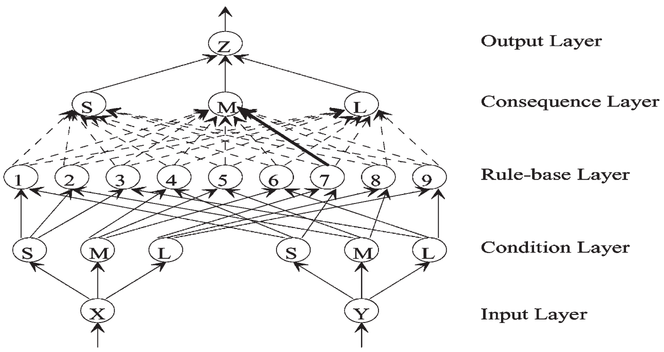


Fig. 6. Sample POPFNN with two input linguistic variables and one output variable.

In POPFNN, once the membership functions are identified through self-organization, each neuron in the condition and the consequence layer represents a label of a linguistic variable. Consider an example to train the POPFNN to identify the fuzzy rules mapping from space $D1$ and $D2$ to $D3$ with two input linguistic variables x and y and an output linguistic variable z where $x \in D1$, $y \in D2$, and $z \in D3$. After the self-organization, each linguistic variable has a label set of {small, median, large}. This set of fuzzy labels shall henceforth be abbreviated as {S, M, L} in the paper. The proposed POP learning algorithm considers all the possible rules, as shown in Fig. 6. There are a total of nine possible rules, each of which is fully connected to the output-label nodes in the consequence layer.

Once the membership functions have been determined, the set of training data is fed into both the input and output layer simultaneously. That is, given the rth set of training data $(x^{(r)}, y^{(r)}; z^{(r)})$, we present $x^{(r)}$ and $y^{(r)}$ at the input layer. The membership values of the input-label nodes are derived using the membership functions that have been determined in phase one. These are subsequently used to produce the firing strength f_k^3 of the kth rule node in the rule-base layer. To obtain the membership values at the output-label nodes, $z^{(r)}$ is fed into the output layer. If the training data satisfy the rule "if x is large and y is small, then z is median," then the derived membership will have large values at the corresponding input-label nodes, namely the input-label node "L" for the linguistic variable "X," the input-label node "S" for the linguistic node "Y," and the output-label node "M" for the defuzzification node "Z." At the same time, the membership values of other input-label (output-label) nodes in the condition (consequence) layer will be relatively small. Hence, the firing strength of the rule, which has the label "L" for the input variable "X" and the label "S" for the input variable "Y" as its conditions, will be larger than the other rules. If the weights of the links that connect the rule nodes and the output-label nodes are raised by the product of the corresponding firing strengths and membership functions, then the improvement in the strength of the link between the rule node having conditions "X is L" and "Y is S" and

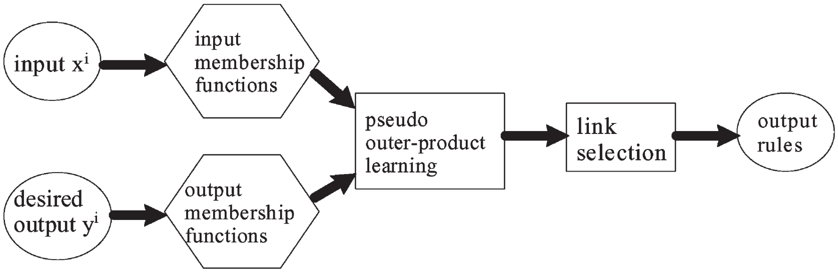

input x^i
input membership functions
desired output y^i
output membership functions
pseudo outer-product learning
link selection
output rules

Fig. 7. Using POP learning algorithm to identify rules.

consequence "Z is M" will unquestionably surpass those of other links. This is shown as a solid line in Fig. 6. As a result, when all the training data are derived from certain fuzzy rules, the links denoting these rules will have larger weights than others. This indicates the existence of such rules in the set of training data. From the discussion, it can be observed that the idea behind POP is quite similar to that of Hebbian's learning algorithm, which is stated as follows: "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently take place in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased" [37]. The Hebbian learning idea is formulated in the proposed POP learning formula as follows:

$$w_{k,l,m} = \sum_{r=1}^{p} f_k^3 \mu_{l,m}\left(y_m^{(r)}\right) \tag{6}$$

where $w_{k,l,m}$ is the weight of the link between the kth rule node and the (l,m)th output-label node in the consequence layer, f_k^3 is the firing strength of the kth rule node given by (3) when presented with the set of training data $x^{(r)}$, $\mu_{l,m}$ is the membership value derived at the (l,m)th output-label node when presented with the set of training data $y^{(r)}$, and $y_m^{(r)}$ is the mth element of the training data $y^{(r)}$.

The weights of the links between the rule-base layer and the output layer are initially set to zero. After the proposed POP learning process, these weights represent the strengths of the fuzzy rules having the corresponding output-label nodes as their real consequences. Among the links between a rule node and all the output-label nodes of a defuzzification node, at most one link with the highest weight is chosen and the others are deleted. On the other hand, when all the weights of the links between a rule node and the output-label nodes of an output linguistic node are very small or almost equal to each other, this means that this rule node has little or no relation to that particular output variable. Hence, all the corresponding links in this case can be deleted without affecting the outputs. When all the links between a rule node and output-label nodes in the consequence layer are removed, this implies that the rule represented by that rule node is irrelevant and should therefore be eliminated. The whole POP learning process is shown in Fig. 7.

VI. Experimental Results and Analysis

Validation of the performance of POPFNN-TVR against FFBP was conducted thoroughly by assessing their respective prediction capabilities on traffic density. This parameter is our main interest, because it is the only macroscopic characteristic that can give a clear indication of the quality of the traffic flow. Different objectives were set prior to each set of experiments involving the prediction of density for all three lanes at site 29. The first aim is to analyze the performance of POPFNN-TVR and FFBP in short-term traffic prediction, whereas the second is to investigate the noise tolerance of both architectures. The speed and density of each lane are predicted using the time, speed, and density information from the remaining two lanes.

A. Measuring Performance

A fundamental concern in prediction and forecasting is to measure how accurate the predicted data set models the desired output data set. Accuracy can be measured using various forms of statistical indicators, such as mean square error (mse), Pearson correlation coefficient (Pearson r), coefficient of determination (R^2), or normalized integral square error (NISE). These indicators measure the error between the predicted and desired data or determine the correlation between them. In this project, the coefficient of determination R^2 is used to benchmark the performance of POPFNN-TVR against that of FFBP.

The most widely used type of correlation coefficient is Pearson r, which is also known as linear or product-moment correlation. Generally, the term correlation can be perceived as the extent to which the values of two variables are "proportional" or linearly related to each other. An interesting property is that the value of the correlation, which is commonly known as the correlation coefficient, does not depend on the specific measurement units used. Correlation between variables is considered high if it can be approximated by a straight line, which is called the regression line or least squares line because it is calculated so as to minimize the sum of the squared distances of all data points from the line. The slope of the regression line range from -1.0 to $+1.0$, which represent the perfect negative and positive correlation coefficients, respectively. Conversely, a coefficient value of zero infers that there is no correlation at all between the variables. Pearson correlation assumes that the two variables are measured on at least interval scales. The Pearson product-moment correlation coefficient r is given as follows:

$$r = \frac{\left[\sum_{t=1}^{T}(y_t - \overline{y})(\hat{y}_t - \overline{\hat{y}})\right]}{\sqrt{\sum_{t=1}^{T}(y_t - \overline{y})^2 \sum_{t=1}^{T}(\hat{y}_t - \overline{\hat{y}})^2}} \tag{7}$$

where y_t is the desired output, $\hat{y}_t$ the computed or predicted output, $\overline{y}$ the mean of desired output y_t, and $\overline{\hat{y}}$ the mean of computed output $\hat{y}_t$.

The coefficient of determination R^2 can be thought of as a measure of linear association between y_t and $\hat{y}_t$ and therefore as a measure of the goodness of fit, with $0 \leq R^2 \leq 1$. When R^2 equals zero, the predicted and desired outputs are totally uncorrelated. In contrast, when R^2 equals 1, the predicted and desired outputs are exactly the same. For the purpose of this project, the alternative formulation of R^2 given as follows was employed:

$$R^2 = \frac{\left[\sum_{t=1}^{T}(y_t - \bar{y})(\hat{y}_t - \bar{\hat{y}})\right]^2}{\sum_{t=1}^{T}(y_t - \bar{y})^2 \sum_{t=1}^{T}(\hat{y}_t - \bar{\hat{y}})^2}. \tag{8}$$

B. Short-Term Traffic Forecasting

The experiments to assess the forecasting ability of POPFNN-TVR and FFBP are organized into two parts. We first assessed the ability of both systems in short-term traffic density prediction from 5 min ahead $(t + 5)$ in time to 1 h ahead $(t + 60)$ and then their ability to tolerate white noise of 10% and 30% induced into the testing data. In all the experiments performed with FFBP, a three-layer feedforward network with 100 neurons in the hidden layer was employed, with a standard sigmoid as for the activation function. This configuration was derived from similar experiments and was by no means the optimal configuration arising from this project. As for POPFNN-TVR, an initial assumption was made that all input data to the network are classifiable into at least three clusters (e.g., low, medium, and high). This assumption is required by the data clustering process that automatically derives the fuzzy sets used for logical inference.

C. Traffic Density Prediction

The aim of this set of experiments is to compare the short-term prediction ability of POPFNN-TVR against that of FFBP. The R^2 index was used to gauge the performance of both networks. Note that different performance criteria may be used, depending on the application. In the forecasting of foreign exchange rate, for instance, the neural network's ability to capture the market trend is far more important than achieving an accurate prediction that would miss most of the upward and downward trends. Experiments were conducted for each of the three lanes at site 29, where time, speed, volume, and density were used as inputs to predict the traffic density for the next 5 to 60 min. Theoretically, there should be a significant correlation between the current density of traffic flow and the density 5 min ahead under normal flow condition, baring, of course, unforeseen events such as accidents. As the time interval gets larger, the correlation should decrease, and thus, the modeling ability of both POPFNN-TVR and FFBP should also decrease. Because POPFNN-TVR has the ability to extract rules from the training set, it will be able to formalize the relations between time and the other input variables, effectively reducing the fluctuation of the predicted output as the forecasting time interval is increased. If these points can be verified from the obtained results, then it can be concluded that POPFNN-TVR possesses greater prediction ability than a conventional FFBP.

Experimental Setup: Two series comprising an identical number of experiments were designed for the FFBP and POPFNN-TVR networks. The problem at hand must first be mapped onto a representative form of the analysis technique, which involves the choice of a suitable neural network architecture. Past research has shown that a multilayer feedforward network, when coupled with a suitable learning algorithm such as the backpropagation algorithm, can be trained to compute any arbitrary nonlinear function [38].

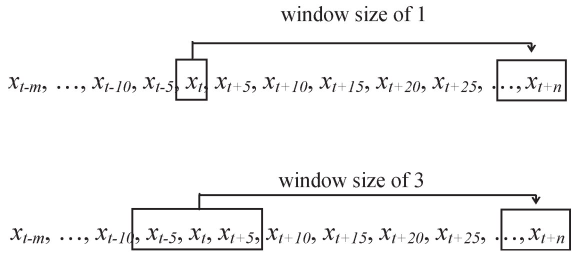

window size of 1
$x_{t-m}, \ldots, x_{t-10}, x_{t-5}, x_t, x_{t+5}, x_{t+10}, x_{t+15}, x_{t+20}, x_{t+25}, \ldots, x_{t+n}$
window size of 3
$x_{t-m}, \ldots, x_{t-10}, x_{t-5}, x_t, x_{t+5}, x_{t+10}, x_{t+15}, x_{t+20}, x_{t+25}, \ldots, x_{t+n}$

Fig. 8. Prediction using time delay windows of one and three, respectively.

FFBP: To gauge the performance of FFBP, time, volume, speed, and density of all three lanes are used as inputs to predict the traffic density of lane 1, lane 2, and lane 3. The input data at time t is used to predict density at $(t + X)$, where X denotes the time interval ranging from 5 to 60 min. In this setup, the length of the time window is 1, as depicted by the top diagram in Fig. 8. As such, the network does not rely on a sequence of past information for its prediction. The bottom diagram in Fig. 8 shows another experimental setup where three consecutive time-dependent variables x_t $(t = t, t - 5$, and $t - 10)$ are used to predict the value of the variable at a future time $t + 25$, hence, the length of the time delay window is 3. The latter technique is normally employed when the only feature vector available is a historical time series of the data to be modeled, as, for instance, in the prediction of the stock market using historical data of shares.

The neural network consists therefore of ten input neurons, which are the time vector and macroscopic parameters (speed, volume, and density) for each of the three lanes, and one output neuron for the density vector to be predicted at $(t + X)$. For the hidden layer, a total of 100 neurons were chosen. Lastly, the sigmoid function was used as the activation function of the network. Instead of the standard backpropagation algorithm, an alternative variant known as backpropagation with momentum term and flat spot elimination was used in these experiments, as this algorithm is known to converge faster and is provided directly by the Stuttgart neural network simulator (SNNS) platform used for implementing FFBP [39]. Four learning parameters are defined as follows: 1) η is the learning parameter, which is used to specify the step width of the gradient descent algorithm; 2) μ is the momentum term, which is used to specify the amount of the old weight change that is added to the current change; 3) c is the constant flat spot elimination value, which is to be added to the derivative of the activation function to enable the network to pass flat spots in the error surface; 4) $d_{\max}$ is the maximum difference $d_j = t_j - o_j$ between a teaching value t_j and an output o_j of an output unit that is propagation back.

After a series of trial experimentation, we found that the network could converge after a reasonable period of time when using the following settings: $\eta = 0.2$, $\mu = 0.5$, $c = 0.1$, and

TABLE III
MAXIMUM, MINIMUM, AND AVERAGE VALUE OF DATA SET FOR SITE 29

	Speed (km/h)			Volume (veh)			Density (veh/km/lane)		
	U1	*U2*	*U3*	*V1*	*V2*	*V3*	*D1*	*D2*	*D3*
Min	0	20	0	0	1	0	0	0.38	0
Max	123.18	115	155	130	150	194	47.40	47.40	37.55
Mean	62.24	77.34	84.13	53.26	57.53	69.81	10.86	9.37	9.93
Std dev	8.69	8.62	21.67	32	28	50	7.31	5.25	18.57
CV (%)	13.96	11.15	11.15	61	49	72	67.31	56.03	63.07

$d_{\max} = 0.0$. No change was made to the network in subsequent experiments other than adjusting the learning rate because no single value is optimal for all dimensions. Having identified the network architecture and learning algorithm, the last stage of the experimental setup involved preprocessing of the input data, the type which is largely dependent on the chosen activation function of the neural network.

The transfer function of a unit is typically chosen so that it can accept input in any range and produce output in a strictly limited range. Although the input can be of any value, the unit is only sensitive to input values within a fairly limited interval. A sigmoid function is used in the experiments, with the steepness factor β set to 1. In this case, the output is in the range [0,1], and the input is sensitive in a range not much larger than $[-1, +1]$. The data samples used for training and testing the neural network were normalized using the min–max normalization technique given as follows:

$$X_n = \frac{(X - X_{\min})}{(X_{\max} - X_{\min})} \tag{9}$$

where X_n is the normalized value, and X, $X_{\min}$ and $X_{\max}$ are an instance of the minimum and the maximum values of the vector to be normalized. This reduces the possibility of reaching the saturation regions of the sigmoid transfer function during training.

Table III summarizes the statistical indicators for the macroscopic parameters at site 29, including the max, min, mean, standard deviation, and coefficient of variance. Using these values, the normalization of data samples is performed by simply substituting the necessary parameters into (9). The normalized data samples are then divided such that 40% will be used for training the neural network, and the other 60% will be used for testing the trained network. It is necessary to ensure that the network is trained using sufficient data to cover the entire problem domain, including maximum and minimum values for each variable as well as a good distribution of values within this range. Selecting a smaller portion of the training data prevents overtraining of the network and ensures that it will generalize well when presented with new data. This criterion for data partitioning is applicable to both FFBP and POPFNN-TVR systems.

POPFNN-TVR: For the case of POPFNN-TVR, the prediction problem is mapped onto a five-layer POPFNN network with the following configuration. The time input is discretized into four labels (early morning, morning, afternoon, night). The normalized traffic volume inputs of the three lanes, which are denoted by $V1$, $V2$, and $V3$, respectively, are represented by three labels (low, medium, high) and, similarly, for the speed and density of the traffic flow, which are denoted by denoted $U1, U2$, and $U3$ and $D1$, $D2$, and $D3$. Lastly, the output density of lane l at $(t + X)$ minutes is denoted by $Dl - X$ discretized into three labels (low, medium, and high).

The optimal number of rule nodes to be used is automatically determined by the LazyPOP algorithm during the unsupervised learning phase. Experimentation with the POPFNN-TVR network employed another form of normalization technique due to the characteristics of the fuzzy neural network. Specifically, because the self-organization phase of POPFNN attempts to learn the structure of the data through clustering, the use of the min–max normalization technique might cause a spurious dispersion of the clusters in the original data, greatly affecting the precision of the data. Thus, the data for POPFNN are normalized in the following manner: Time is normalized to 2400 h, U_i are normalized to 80 km/h, whereas V_i and D_i are normalized to the average volume and density of the ith lane, respectively. This method allows the normalized speed to be interpreted more easily as compared to the min–max technique.

Results and Analysis: In this phase, seven experiments had to be carried out for the first lane of site 29 and five experiments for the two remaining lanes. Therefore, a total of 17 experiments were conducted using FFBP and, similarly, using POPFNN-TVR. Due to the large number of experiments and the long training time required, the training of the FFBP network was executed in batch mode using SNNS. Batch processes were programmed to terminate when the SSE expressed in (10) reached 0.8 or less:

$$\text{SSE} = \sum_{i=1}^{T} (\hat{y}_i - y_i)^2. \tag{10}$$

The results of FFBP predictions for all the three lanes were recorded for increasing values of the time interval ranging from $t + 5$ to $t + 60$ min. The time samples spanned over 2400 min for a period of approximately 3 days from September 7 to 10, 1996. The computed output of the network was plotted against the desired output to give an indication of FFBP's ability to follow the general trend of the traffic data, and the R^2 values were calculated accordingly. Similarly, experimental data were collected using POPFNN-TVR for each of the three lanes, its outputs monitored, and the R^2 values calculated accordingly. In general, traffic flow follows a repetitive pattern, even though its amplitude is stochastic in nature, and so two near-periodic cycles were observed. Specifically, the traffic density was low at the beginning of the day and showed significant increase on two occasions, namely 1) the morning peak hours P_1 when large

TABLE IV
R^2 OF LANE 1 PREDICTION ($D1$)

Time Interval	R^2 (D1-FFBP)	Variation(%)	R^2 (D1-TVR)	Variation(%)
5	0.615254	*N.A.*	0.798041	*N.A.*
10	0.483418	21%	0.762742	4%
15	0.470706	3%	0.757262	1%
20	0.516054	10%	0.751977	1%
25	0.467644	9%	0.712639	5%
45	0.412187	12%	0.6466	9%
60	0.421244	2%	0.607852	6%
Average	0.483786	10%	0.719587	4%

TABLE V
R^2 OF LANE 2 PREDICTION ($D2$)

Time Interval	R^2 (D2-FFBP)	Variation(%)	R^2 (D2-TVR)	Variation(%)
5	0.219135	*N.A.*	0.718982	*N.A.*
10	0.438324	100%	0.70406	2%
15	0.215074	51%	0.688784	2%
20	0.058599	73%	0.647188	6%
25	0.045363	23%	0.664814	3%
Average	0.195299	62%	0.684765	3%

TABLE VI
R^2 OF LANE 3 PREDICTION ($D3$)

Time Interval	R^2 (D3-FFBP)	Variation(%)	R^2 (D3-TVR)	Variation(%)
5	0.644698	*N.A.*	0.760657	*N.A.*
10	0.487176	24%	0.811066	7%
15	0.148185	70%	0.804638	1%
20	0.126936	14%	0.776164	4%
25	0.041129	68%	0.760657	2%
Average	0.289624	44%	0.7826364	3%

pools of people are traveling to work and 2) the evening peak hours P_2 when commuters are traveling back home. Unless unforeseen circumstances arise, this trend is similarly observed during weekdays.

The FFBP network was able to model the general trend of the traffic flow to a certain extent. However, the quality of the prediction (goodness of fit) was poor, as evidenced by its inability to model the evening peak traffic. In addition, severe overshoot was observed in the prediction for the first time period, as indicated by the large gap between the predicted and desired densities during that particular period of time. For lane 3, the inability to model peak traffic condition was even more severe, especially with the increase in prediction interval. This phenomenon, which is observed both during the morning as well as the evening peak hours, could be due to the high-density fluctuation in lane 3 traffic, which impaired FFBP's ability to model these peaks. In comparison, the prediction ability of POPFNN-TVR was significantly better. The R^2 values computed over the increasing time interval for both FFBP and POPFNN-TVR are summarized in Table IV in columns 2 and 4, respectively. Last, the variations in R^2 with respect to the R^2 value obtained in the last prediction interval are shown in columns 3 and 5 of the same table. Likewise, Tables V and VI show the R^2 values and its fluctuations obtained for lanes 2–3.

It can be observed from Tables IV–VI that the R^2 values obtained from POPFNN-TVR are much higher compared to the values obtained using FFBP. Specifically, these values range on the average from 0.6 to 0.8 for POPFNN and from 0.04 to 0.64 for FFBP, which means that POPFNN is able to compute the density much more accurately than the FFBP network. In addition, the fluctuation in R^2 for POPFNN-TVR is low, hardly exceeding 9% over the entire prediction time intervals from $(t + 5)$ to $(t + 60)$. In contrast, the variations for FFBP are much higher, with the majority exceeding 10% for the three lanes. This fluctuation is extremely severe in lane 2 with a maximum variation of 100%. In addition, R^2 is also maintained across the predictions for all the three lanes used. Plotting the R^2 fluctuation graphs for both FFBP and POPFNN-TVR shows that the R^2 values obtained using the latter have a much smoother descent when compared against FFBP. This implies that POPFNN-TVR is able to maintain consistent performance even with large prediction intervals, whereas the performance of FFBP drops drastically as prediction intervals increase.

Further experiments show that a higher percentage of predicted density falls within the qualifying bands described in earlier paragraphs. On the average, the ratio of predictions that are within range for POPFNN-TVR over FFBP is greater than 1.5. This observation further substantiates our initial hypothesis that POPFNN-TVR provides a better prediction model compared to FFBP. Overall, two conclusions can be made from the results obtained. First, the comparison of the R^2 values in Tables IV–VI proves that the short-term prediction ability of POPFNN-TVR is superior to that of FFBP. Second, the low fluctuation in R^2 evidenced in the same tables also suggests that POPFNN-TVR is a more suitable model for prediction over longer period of time.

D. Noise Tolerance Ability

The next set of experiments serves to study and contrast the networks' noise tolerance ability, with the objective to prove that POPFNN-TVR is more tolerant to noisy input data than FFBP. To verify this hypothesis, POPFNN-TVR must be able to maintain its prediction capability under the influence of noisy input data. The performance of POPFNN-TVR is again gauged using the R^2 indicator. The test data used in the previous section have to be preprocessed prior to the experiments, with white noise of 10% and 30% injected into the volume V_i, speed U_i, and density D_i components of the original test set, whereas the time and desired output density are left unchanged for comparison purposes. In the experiments described here, POPFNN-TVR only operated in the test mode, which means that the network performs pure forward propagation of the input data without modification. The preprocessed noisy data are fed into POPFNN-TVR, and the corresponding outputs are obtained at the output layer, which are then compared against the desired outputs to generate the performance indexes. A similar set of experiments is performed using FFBP and the final computed R^2 values are compared against that obtained for POPFNN-TVR.

Experimental Setup: The setup for this series of experiments is similar to that conducted previously. The network structure and the weights of all synaptic links for both FFBP and POPFNN-TVR networks as trained in the previous experiments were saved to facilitate further experimentation. The structures of the two networks are given as follows. The FFBP consists of three layers, namely 1) the input layer with ten neurons (time, $V1$, $V2$, $V3$, $U1$, $U2$, $U3$, $D1$, $D2$, and $D3$), 2) the hidden layer with 100 neurons, and 3) the output layer with one neuron ($Dl - X$, which is the lane l density at $t + X$). POPFNN-TVR, on the other hand, is configured with four input variables, with the time discretized into four labels (early morning, morning, afternoon, and night) and V, U, D each discretized into three labels (low, medium, and high), and one output variable $Dl - X$, which is again discretized into three labels (slow, medium, and high).

The new test sets are generated by introducing white noise of 10% and 30% into the original test data used in the previous section. At each noise level, seven test sets are generated for lane 1 and five for each of the remaining two lanes. In total, 34 new test sets are generated for each of the two networks. Fig. 9 shows a segment of the test data perturbed using 30% white noise where the original data are shown as dotted line and the noisy data as solid line. The vertical axis represents the normalized lane 1 density, and the horizontal axis represents the data collection time at 5-min intervals. This set is used for predicting the lane 1 density at $(t + 5)$ using POPFNN-TVR. It should be noted that the 30% white noise refers to a random signal within the ±15% band of the original signal. By summing up the noise and the original signal, a new test set is obtained. The response of both FFBP and POPFNN-TVR are studied by analyzing the computed outputs of these networks when noisy data are fed.

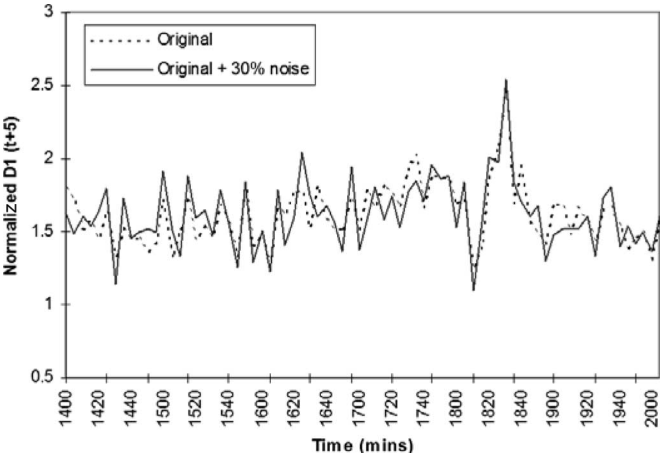


Fig. 9. Test data with and without 30% white noise perturbation.

Results and Analysis: The results obtained from these experiments are summarized in Tables VII and VIII, which show the variation in R^2 when the two networks are subject to noisy input data perturbed using 10% and 30% white noise. These values are compared against the R^2 values obtained in the previous section to give an indication of how FFBP and POPFNN-TVR cope with the noise. Numbers in columns 5 and 8 of both tables show the variation in R^2 for each of the experiments conducted using FFBP and POPFNN-TVR. The average (absolute) fluctuations of each prediction variable expressed in percentages are listed as well.

From Tables VII and VIII, it can be observed that there exists a general downward trend in the new R^2 compared against its original value. This trend is in line with the general assumption that the performance of the system should be inversely proportional to the amount of noise in the input data. However, there are cases where the system performance improved as a result of adding noise. These are characterized in Tables VII and VIII as positive values for the R^2 fluctuations. This, however, does not imply that the respective network will perform better than the other under the influence of noise. A more accurate measure of performance would be the absolute degree of variations in R^2, i.e., without taking the sign into consideration. The absolute average is expressed as

$$\text{Average}(\text{absolute}) = \frac{\sum_{i=1}^{\text{n}} |\text{Diff}(\%)_i|}{n} \tag{11}$$

TABLE VII
R^2 FLUCTUATION WITH 10% WHITE NOISE

Prediction Variable	Time Interval	FFBP			POPFNN-TVR		
		R^2 (10%)	R^2 (Org)	Diff (%)	R^2 (10%)	R^2 (Org)	Diff (%)
D1	5	0.611904	0.615254	-0.54%	0.799398	0.798041	0.17%
	10	0.481628	0.483418	-0.37%	0.760602	0.762742	-0.28%
	15	0.466912	0.470706	-0.81%	0.756172	0.757262	-0.14%
	20	0.515324	0.516054	-0.14%	0.750022	0.751977	-0.26%
	25	0.466951	0.467644	-0.15%	0.710375	0.712639	-0.32%
	45	0.410819	0.412187	-0.33%	0.651819	0.6466	0.81%
	60	0.421994	0.421244	0.18%	0.607698	0.607852	-0.03%
AVERAGE (absolute)				0.36%			0.28%
D2	5	0.214877	0.219135	-1.94%	0.722283	0.718982	0.46%
	10	0.473931	0.438324	8.12%	0.697704	0.70406	-0.90%
	15	0.039899	0.215074	-81.45%	0.678566	0.688784	-1.48%
	20	0.002690	0.058599	-95.41%	0.644236	0.647188	-0.46%
	25	0.242888	0.045363	435.43%	0.66286	0.664814	-0.29%
AVERAGE (absolute)				124.47%			0.72%
D3	5	0.640380	0.644698	-0.67%	0.846645	0.760657	11.30%
	10	0.481102	0.487176	-1.25%	0.80101	0.811066	-1.24%
	15	0.145965	0.148185	-1.50%	0.802056	0.804638	-0.32%
	20	0.126122	0.126936	-0.64%	0.774006	0.776164	-0.28%
	25	0.040529	0.041129	-1.46%	0.763084	0.760657	0.32%
AVERAGE (absolute)				1.1%			2.69%

TABLE VIII
R^2 FLUCTUATION WITH 30% WHITE NOISE

Prediction Variable	Time Interval	FFBP			POPFNN-TVR		
		R^2 (30%)	R^2 (Org)	Diff (%)	R^2 (30%)	R^2 (Org)	Diff (%)
D1	5	0.612565	0.615254	-0.44%	0.783441	0.798041	-1.83%
	10	0.477121	0.483418	-1.30%	0.749518	0.762742	-1.73%
	15	0.455773	0.470706	-3.17%	0.736683	0.757262	-2.72%
	20	0.498322	0.516054	-3.44%	0.74175	0.751977	-1.36%
	25	0.468988	0.467644	0.29%	0.700727	0.712639	-1.67%
	45	0.400731	0.412187	-2.78%	0.62671	0.6466	-3.08%
	60	0.413233	0.421244	-1.90%	0.583465	0.607852	-4.01%
AVERAGE (absolute)				1.9%			2.34%
D2	5	0.211386	0.219135	-3.54%	0.707836	0.718982	-1.55%
	10	0.456446	0.438324	4.13%	0.749518	0.70406	6.46%
	15	0.039171	0.215074	-81.79%	0.363816	0.688784	-47.18%
	20	0.01451	0.058599	-75.24%	0.630061	0.647188	-2.65%
	25	0.224301	0.045363	394.46%	0.650099	0.664814	-2.21%
AVERAGE (absolute)				111.83%			12.01%
D3	5	0.617079	0.644698	-4.28%	0.841662	0.760657	10.65%
	10	0.461799	0.487176	-5.21%	0.802574	0.811066	-1.05%
	15	0.139921	0.148185	-5.58%	0.775247	0.804638	-3.65%
	20	0.11191	0.126936	-11.84%	0.77506	0.776164	-0.14%
	25	0.034848	0.041129	-15.27%	0.754831	0.760657	-0.77%
AVERAGE (absolute)				8.44%			3.25%

where Diff(%) are the values listed in columns 5 and 8 of Tables VII and VIII, and n is the total number of experiments for each prediction variable.

The values thus calculated for this measure are denoted as the absolute average in Tables VII and VIII. A lower fluctuation in R^2 signifies that the network is able to prune spurious noise while maintaining its original characteristic. In contrast, a high value indicates a large deviation in the new R^2 from the original R^2. In Table VIII, it can be observed that R^2 fluctuation for POPFNN-TVR is generally much lower than FFBP across all three lanes. The fluctuation in the prediction of the lane 2 density is particularly serious for the FFBP network, as R^2 variation can reach a high of over 400% in certain cases. However, it is observed that the absolute average of the lane 3 prediction

TABLE IX
R^2 FLUCTUATION WITH WHITE NOISE INCREASED FROM 10% TO 30%

Prediction Variable	FFBP			POPFNN-TVR		
	AveDiff (10% noise)	*AveDiff (30% noise)*	Diff (%)	*AveDiff (10% noise)*	*AveDiff (30% noise)*	Diff (%)
D1	0.36	1.9	1.54	0.28	2.34	2.06
D2	124.47	111.83	-12.64	0.72	12.01	11.29
D3	1.1	8.44	7.34	2.69	3.25	0.56
Average (absolute)			7.17			4.64

using POPFNN-TVR is 2.69%, which is higher than the 1.1% obtained using FFBP. This is due to the higher fluctuation in R^2 for predicting density at $(t+5)$. On the whole, the fluctuation for $D3$ is still much lower when compared against that of FFBP.

As the noise level increases, the fluctuations in R^2 for both networks also increase as a result. A comparison between the Diff(%) values in Tables VII and VIII was performed, the results of which are summarized in Table IX. This table clearly shows the variation in R^2 when noise level is raised from 10% to 30%. Columns 1 and 2 represent the average percentage variations in R^2 for FFBP, when white noise of 10% and 30% noise are introduced. Columns 5 and 6 provide the same information for POPFNN-TVR. The difference between these two values is expressed in (12) and is listed in columns 4 and 7:

$$\text{Diff} = \text{AveDiff}(30\%\ \text{noise}) - \text{AveDiff}(10\%\ \text{noise}). \qquad (12)$$

Last, the absolute average given in the last row of Table IX is computed as given in (11). It can be observed that the average fluctuation for the prediction variable $D2$ using FFBP has decreased by 12.64% (from 124.47% to 111.83%), whereas that of POPFNN-TVR has increased by 11.29% (from 0.72% to 12.01%). This result is due to the huge improvement in the lane 2 prediction at $(t+25)$, which leads to the lower average fluctuation for FFBP as shown in the respective rows in Tables VII and VIII. Despite the lower average figure, the overall fluctuation in predicted density using FFBP for the other lanes follows an upward trend. On the other hand, the fluctuation in R^2 for POPFNN-TVR is more gradual over the three lanes, which explains why a decrease in average fluctuation is not observed. On the average, though, the fluctuation in R^2 for the POPFNN-TVR is still much lower than similar figures obtained for FFBP. This result can be highlighted by contrasting the figures listed in columns 5 and 8 in Table VIII.

This comparison demonstrated that POPFNN-TVR is more capable in pruning noisy data and maintaining a reasonable level of prediction with respect to the desired outputs. Therefore, it can be concluded that POPFNN-TVR has better noise tolerance ability compared to FFBP. However, it should be noted that there is a limit to this capability. Specifically, if the noise level is increased further to 40%, 50%, etc., then at some point, both FFBP and POPFNN-TVR networks will become unstable and their predicted outputs no longer reliable.

VII. CONCLUSION

This paper has described a novel approach to traffic flow analysis and prediction using a specific class of fuzzy neural network known as the POPFNN-TVR. The system was successfully trained and subsequently used for short-term traffic flow prediction. Experimental results have demonstrated the ability of POPFNN-TVR to extract a valid set of fuzzy rules from the training data as well as to generalize and react appropriately to new input data. Comparative analysis has demonstrated that POPFNN-TVR possesses a higher degree of prediction capability than a conventional feedforward neural network using backpropagation learning. The proposed approach thus not only facilitates the understanding and analysis of traffic control problems by allowing a semantically rich set of rules to model system behavior but also offers better performance and noise tolerance as compared to other approaches.

The research presented in this paper is in line with the research direction of the Centre for Computational Intelligence (C2i), formerly Intelligent Systems Laboratory at the Nanyang Technological University, Singapore. C2i undertakes active research in intelligent neural fuzzy systems for the modeling of complex, nonlinear, and dynamic problem domains. Examples of neural and fuzzy neural systems developed at C2i are MCMAC [40], GenSoFNN [23], RSPOP [30], and POPFNN [14]. These have been applied to novel and interesting applications such as automated driving [41], signature forgery detection [42], gear control for automotive continuous variable transmission (CVT) [12], pediatric cancer subtype classification [43], and fingerprint verification [44].

REFERENCES

[1] M. S. Dougherty and H. R. Kirby, "The use of neural networks to recognize and predict traffic congestion," *Traffic Eng. Control*, vol. 34, no. 6, pp. 311–314, 1994.

[2] C. Ledoux, "An urban traffic flow model integrating neural network," *Transp. Res.*, vol. 5C, no. 5, pp. 287–300, Oct. 1997.

[3] H. Adeli and A. Karim, "Fuzzy-wavelet RBFNN model for freeway incident detection," *J. Transp. Eng.*, vol. 126, no. 6, pp. 464–471, Nov. 2000.

[4] S. Ishak and H. Al Deek, "A fuzzy art neural network model for automated detection of freeway incidents," *Transp. Res. Rec.*, vol. 1634, no. 1634, pp. 56–63, 1998.

[5] H. Xu, C. M. Kwan, L. Haynes, and J. D. Pryor, "Real-time adaptive online traffic incident detection," in *Proc. IEEE Int. Symp. Intell. Control*, 1996, pp. 200–205.

[6] J. Chou, W. A. O'Neil, and H. Cheng, "Pavement distress evaluation using fuzzy logic and moment invariants," *Transp. Res. Rec.*, vol. 1505, no. 1505, pp. 39–46, 1995.

[7] H. Yin, S. C. Wong *et al.*, "Urban traffic flow prediction using a fuzzy-neural approach," *Transp. Res.*, vol. 10, no. 2, pt. C, pp. 85–98, Apr. 2002.

[8] Z. Yuan, W. Li, and H. Liu, "Forecast of dynamic traffic flow," in *Proc. ICTTS*, 2000, pp. 507–512.

[9] K. K. Ang, C. Quek, and A. Wahab, "MCMAC-CVT: A novel associative memory based CVT transmission control system," *Neural Netw.*, vol. 15, no. 3, pp. 219–236, Mar. 2002.

[10] K. Hayashi, Y. Shimizu, Y. Dote, A. Takayama, and A. Hirako, "Neuro fuzzy transmission control for automobile with variable loads," *IEEE Trans. Control Syst. Technol.*, vol. 3, no. 1, pp. 49–52, Mar. 1995.

[11] H. Xiao, H. Sun, and B. Ran, "The fuzzy-neural network traffic prediction framework with wavelet decomposition," *Transp. Res. Rec.*, to be published.

[12] A. Kotsialos, M. Papageorgiou, C. Diakaki, Y. Pavis, and F. Middelham, "Traffic flow modeling of large-scale motorway using the macroscopic modeling tool METANET," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 4, pp. 282–292, Dec. 2002.

[13] C. H. Wu, J. M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[14] R. W. Zhou and C. Quek, "POPFNN: A pseudo outer-product based fuzzy neural network," *Neural Netw.*, vol. 9, no. 9, pp. 1569–1581, Dec. 1996.

[15] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1. Cambridge, MA: MIT Press, 1986.

[16] C. T. Lin, *Neural Fuzzy Control System With Structure and Parameter Learning*. Singapore: World Scientific, 1994.

[17] C. T. Lin and C. S. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput.*, vol. 40, no. 12, pp. 1320–1336, Dec. 1991.

[18] R. R. Yager, "Modeling and formulating fuzzy knowledge bases using neural network," *Neural Netw.*, vol. 7, no. 8, pp. 1273–1283, 1994.

[19] I. Hayashi, H. Nomura, H. Yamasaki, and N. Wakami, "Construction of fuzzy inference rules by NDF and NDFL," *Int. J. Approx. Reason.*, vol. 6, no. 2, pp. 241–266, Feb. 1992.

[20] M. Lee, S. Y. Lee, and C. H. Park, "A new neuro-fuzzy identification model of nonlinear dynamic systems," *Int. J. Approx. Reason.*, vol. 10, no. 1, pp. 30–44, Jan. 1994.

[21] H. Ishibuchi, H. Tanaka, and H. Okada, "Interpolation of fuzzy if-then rules by neural networks," *Int. J. Approx. Reason.*, vol. 10, no. 1, pp. 3–27, Jan. 1994.

[22] C. Quek and R. W. Zhou, "The POP learning algorithms: Reducing work in identifying fuzzy rules," *Neural Netw.*, vol. 14, no. 10, pp. 1431–1445, Dec. 2001.

[23] W. L. Tung and C. Quek, "GenSoFNN: A generic self-organising fuzzy neural network," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1075–1086, Sep. 2002.

[24] J. M. Keller, R. R. Yager, and H. Tahani, "Neural network implementation of fuzzy logic," *Fuzzy Sets Syst.*, vol. 45, no. 1, pp. 1–12, Jan. 1992.

[25] B. Krause, C. von Altrock, K. Limper, and W. Schafers, "A neuro-fuzzy adaptive control strategy for refuse incineration plants," *Fuzzy Sets Syst.*, vol. 63, no. 3, pp. 329–338, May 1994.

[26] S. Horikawa, H. Furuhashi, and Y. Uchikawa, "On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 801–806, Sep. 1992.

[27] S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference systems," *IEEE Trans. Syst., Man, Cybern.*, vol. 5, no. 23, pp. 665–685, May/Jun. 1993.

[28] S. K. Halgamuge and M. Glesner, "Neural networks in designing fuzzy systems for real world applications," *Int. J. Fuzzy Sets Syst.*, vol. 65, no. 1, pp. 1–12, Jul. 1994.

[29] C. Quek and W. L. Tung, "Falcon: Fuzzy neural control and decision systems using FKP and PFKP clustering algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. 34, no. 1, pp. 686–694, Feb. 2004.

[30] K. K. Ang and C. Quek, "RSPOP: Rough set-based pseudo-outer-product fuzzy rule identification algorithm," *Neural Comput.*, vol. 17, no. 1, pp. 205–243, Jan. 2005.

[31] C. Quek and R. W. Zhou, "POPFNN-AARS: A pseudo outer-product based fuzzy neural network," *IEEE Trans. Syst., Man, Cybern.*, vol. 29, no. 6, pp. 859–870, Dec. 1999.

[32] P. J. Werbos, "Backpropagation: Past and future," in *Proc. 2nd Int. Conf. Neural Netw.*, 1988, pp. 343–353.

[33] R. L. Mantaras, *Approximate Reasoning Models*. London, U.K.: Ellis Horwood Limited, 1990.

[34] T. Kohonen, *Self-Organisation and Associative Memory*. Berlin, Germany: Springer-Verlag, 1988.

[35] B. Kosko, "Unsupervised learning in noise," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 44–57, Mar. 1990.

[36] P. J. Werbos, "Neuro-control and fuzzy logic: Connections and designs," *Int. J. Approx. Reason.*, vol. 6, no. 2, pp. 185–219, Feb. 1992.

[37] D. O. Hebb, *The Organisation of Behaviour*. New York: Wiley, 1949.

[38] D. W. Patterson, *Artificial Neural Networks, Theory and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

[39] A. Zell *et al.*, "SNNS: Stuttgart neural network simulator user manual," Institute for Parallel and Distributed High Performance Systems (IPVR), Univ. Stuttgart, Stuttgart, Germany, Tech. Rep. 6, 1995.

[40] K. K. Ang and C. Quek, "Improved MCMAC with momentum neighborhood and averaged trapezoidal output," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 3, pp. 491–500, Jun. 2000.

[41] M. Pasquier, C. Quek, and M. Toh, "Fuzzylot: A self-organising fuzzy-neural rule-based pilot system for automated vehicles," *Neural Netw.*, vol. 14, no. 8, pp. 1099–1112, 2001.

[42] C. Quek and R. W. Zhou, "Antiforgery: A novel pseudo-outer product based fuzzy neural network driven signature verification system," *Pattern Recognit. Lett.*, vol. 23, no. 14, pp. 1795–1816, Dec. 2002.

[43] W. L. Tung and C. Quek, "GenSo-FDSS: A neural-fuzzy decision support system for pediatric ALL cancer subtype identification using gene expression data," *Artif. Intell. Med.*, vol. 33, no. 1, pp. 61–88, Jan. 2005.

[44] C. Quek, K. B. Tan, and V. K. Sagar, "Pseudo-outer product based fuzzy neural network fingerprint verification system," *Neural Netw.*, vol. 14, no. 3, pp. 305–323, Apr. 2001.

Chai Quek received the B.Sc. degree in electrical and electronics engineering and the Ph.D. degree in intelligent control from Heriot Watt University, Edinburgh, U.K., in 1986 and 1990, respectively.

He is an Associate Professor and a Member of the Centre for Computational Intelligence (formerly the Intelligent Systems Laboratory), School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include intelligent control, intelligent architectures, artificial intelligence in education, neural networks, fuzzy systems, fuzzy rule-based systems, genetic algorithms, neurocognitive computational architectures, and semantic learning memory systems.

Michel Pasquier received the B.S. degree in electrical engineering and the Ph.D. degree in computer science from the National Polytechnic Institute of Grenoble, Grenoble, France, in 1985 and 1988, respectively,

In 1989, he left the LIFIA laboratory (now GRAVIR-INRIA) for the Science City of Tsukuba, Japan, where he worked as a Visiting Researcher at the ElectroTechnical Laboratory until 1992. He then joined the Sanyo Electric's Intelligent Systems Laboratory. In 1994, he joined the School of Computer Engineering, Nanyang Technological University, Singapore, where he currently teaches artificial intelligence and other computer science courses. He is also a Co-Founder and the Director of the Centre for Computational Intelligence, Nanyang Technological University. He has led funded projects and served as a Consultant in various application areas, including intelligent robotics and automation, transportation and automotive systems, online learning and e-business, and financial engineering. His research interests include cognitive architectures and systems, adaptation and learning processes, and nature-inspired systems, as well as methods for approximate reasoning, optimization, and planning.

Bernard Boon Seng Lim received the B.Eng. degree in computer engineering from the School of Applied Science, Nanyang Technological University, Singapore, in 1997.

He is currently a Manager at Ernst & Young Laboratories for Internet and Security within the TSRS practice of Ernst & Young, Singapore. Prior to joining Ernst & Young, he was with the DSO National Laboratories Singapore, specializing in system and networking intrusion research and penetration testing for many government ministries and statutory boards in Singapore.