This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

Selection of significant on-road sensor data for short-term traffic flow forecasting using the Taguchi method

Kit Yan Chan, *member IEEE*, S. Khadem, Tharam S. Dillon, *Fellow IEEE,* V. Palade, *senior member IEEE* , J. Singh, *member IEEE,* E.Chang, *senior member IEEE*

***Abstract* — Over the past two decades, neural networks have been applied to develop short-term traffic flow predictors. The past traffic flow data, captured by on-road sensors, is used as input patterns of neural networks to forecast future traffic flow conditions. The amount of input patterns captured by the on-road sensors is usually huge, but not all input patterns are useful when trying to predict the future traffic flow. The inclusion of useless input patterns is not effective to developing neural network models. Therefore, the selection of appropriate input patterns, which are significant for short-term traffic flow forecasting, is essential. This can be conducted by setting an appropriate configuration of input nodes of the neural network; however, this is usually conducted by trial and error. In this paper, the Taguchi method, which is a robust and systematic optimization approach for designing reliable and high-quality models, is proposed for the purpose of determining an appropriate neural network configuration, in terms of input nodes, in order to capture useful input patterns for traffic flow forecasting. The effectiveness of the Taguchi method is demonstrated by a case study, which aims to develop a short-term traffic flow predictor based on past traffic flow data captured by on-road sensors located on a Western Australia freeway. Three advantages of using the Taguchi method were demonstrated: (1) short-term traffic flow predictors with high accuracy can be designed; (2) the development time for short-term traffic flow predictors is reasonable; (3) the accuracy of short-term traffic flow predictors is robust with respect to the initial settings of the neural network parameters during the learning phase.**

***Index Terms* — Taguchi method, neural networks, traffic flow forecasting, sensor data, neural network configuration, input patterns**

I. Introduction

Forecasting accurate traffic flow conditions has long been identified as a proactive approach to regional traffic control [28]. The approach can be broadly classified under: i) short-term and ii) long-term traffic flow forecasting [1]. Long-term forecasting provides monthly or yearly traffic flow forecasting conditions and is commonly used for long-term planning of transportation or construction. Short-term forecasting focuses on making predictions about the likely traffic flow changes in the short-term, typically within ten minutes ahead. It provides traffic forecasting required for traffic operations and control, with a lead time of a few minutes based on traffic flow data, which is captured by a set of on-road sensors installed along the freeway. It also assists the proactive traffic control centre to anticipate traffic congestion and improve the mobility of transportation [37]. This paper focuses on the development of robust and accurate short-term traffic flow predictors, concerned with producing real-time forecasts for a few minutes ahead, after the short-term traffic flow predictor has received past traffic flow data captured from on-road sensors within the past few minutes.

The short-term traffic flow predictor represents a multi-input-single-output system, which relates the past traffic flow conditions to the future traffic flow conditions. Prior to developing the short-term traffic flow predictor, the amount of input patterns of the short-term traffic flow predictor, which represents the amount of past traffic flow conditions captured by on-road sensors, has to be determined. Subsequently, traditional time-series forecasting methods, such as filtering methods [23], moving average methods [29], k-nearest-neighbour methods [7] and Kalman filters [37], can be used to develop a short-term traffic flow predictor. Results show that short-term traffic flow predictors developed by these traditional forecasting methods can achieve reasonable accuracy in predicting future traffic flow conditions, but their ability to capture the strongly non-linear characteristics of short-term traffic flow data is questionable. Also, the determination of the amount of past traffic flow conditions used as input patterns of the models is based on a trial and error method, which is very much time-consuming.

Another commonly used approach, namely neural networks (NNs) [11, 13, 22], has been applied to develop short-term traffic flow predictors [4, 8, 19], in order to address the non-linear traffic flow characteristics. To further enhance the generalization capability of NNs, research has been conducted by incorporating NNs with other computational intelligence methods or statistical prediction methods, such as fuzzy systems [12, 26, 38], Kalman filter [31], fuzzy clustering method [30], and the autoregression moving average method [35] etc.

Even if hybrid NNs can achieve better traffic flow forecasting results than those obtained by using only NNs, the limitation of determining useful input patterns to NNs for traffic flow forecasting is still not resolved. In fact, there are huge amount of input patterns captured by the on-road sensors. Using all input patterns which includes useless input patterns is not most effective in developing NNs. Zhang et al [39], and Lachtermacher and Fuller [18] mentioned that the determination of an appropriate amount of input patterns for the NN is a significant design factor for time-series forecasting. Too many or too few input patterns may significantly affect their forecasting effectiveness. Apart from this, an optimal NN configuration, which includes an optimal number of hidden nodes, is also closely correlated with the amount of input patterns [39]. The specification of appropriate input patterns can be configured with appropriate input nodes of the neural network, which is usually conducted by trial and error. Therefore, it is desirable to develop a methodical approach for determining the appropriate amount of input patterns of short-term traffic flow predictors, so that designers can identify a feasible solution-searching region, in order to improve the quality of traffic flow forecasting.

In quality control, the Taguchi method has been successfully used to design reliable and high-quality products at low cost for various items, such as automobiles and consumer electronics [14, 32]. Based on our observation, the determination of topologies of short-term traffic flow predictors with high accuracy and robustness can be considered as the design of a high quality product, or the design of robust manufacturing processes [21], where both designs aim to produce the resultant functionality which approximates the ideal function as closely as possible. In fact, the Taguchi method has also been used for the design of configurations of NNs for various manufacturing processes [16, 17, 24, 27, 52] and material characteristics [41, 20]. In this paper, we propose the Taguchi method as a means of developing optimal topologies of short-term traffic flow predictors by optimizing the configuration and the inputs of NN models.

The present research, which adopts the Taguchi method to investigate the significance of the design factors for short-term traffic flow predictors, can be classified into two types: a) past traffic flow conditions used as input patterns to the NN, and b) NN configurations, such as the number of hidden nodes, the number of hidden layers, the activation functions between nodes, etc. In accordance with the Taguchi method, these design factors are arranged in an inner orthogonal array. The Taguchi method conducts systematic trials based on orthogonal arrays to study the design factors using a small number of trials, and then it estimates the appropriate values of the design factors that can optimize a given performance measure, typically the differences between the actual data and the responses of the short-term traffic flow predictors. The Taguchi method is intended to achieve the following objectives, which are critical factors for the development of the short-term traffic flow predictors: (1) high accuracy, which is required for short-term traffic flow predictors; (2) reasonable time for the development of short-term traffic flow predictors; and (3) robust accuracy of short-term traffic flow predictors, which can withstand the initial setting of the NN weights during the learning phase, in order to reduce the chance of settling at local minima.

The rest of the paper is organized as follows. Section 2 provides a brief description of the Taguchi method. Section 3 defines and describes the topology of the short-term traffic flow predictor. In Section 4, the main operations of the Taguchi method for short-term traffic flow predictor design, which involve the identification of design factors, the specification of objective functions, the trial design, and analysis of accuracies and reliabilities, are discussed. Finally, the discussion of the results and some conclusions, regarding short-term traffic flow predictor design using the Taguchi method, is presented in Section 5.

II. Robust design using Taguchi method

In this section, the Taguchi method, which has been widely used for the robust design of products [14, 32], is briefly described. To develop a product, the basic functional prototype design, which defines the configuration and attributes of the product undergoing analysis or development, is first initiated based on the designers' knowledge of the product. The initial design is usually far from optimal in terms of quality or robustness. Therefore, it is necessary to identify the settings of design factors that optimize the performance characteristics and minimize the sensitivity of engineering designs to the sources of variation.

A common approach to design optimization is to experiment with the design factors one at time or by the trial and error method, until a reasonable design with certain qualities is found. However, with the trial and error method, it may take a very long time to complete the design optimization. To determine the optimum conditions, a "full factorial" approach can be used, where all possible combinations of levels in all parameters are considered. When a product has n design factors and each of them has k levels, the total number of combinations of levels for the "full factorial" approach is k^n. When the number of design factors is large, it is almost impossible to test all possible combinations of levels in all design factors. For example, if the designer is studying 15 design factors with two levels, a full factorial approach requires examining 32768 $\left(\text{i.e.: } 2^{15}\right)$ experiments.

Table 1 Orthogonal array ($L_9(3)$)

Experiments	Design variable A	Design variable B	Design variable C	Design variable D
1	1	1	1	1
2	1	2	2	2
3	1	3	3	3
4	2	1	2	3
5	2	2	3	1
6	2	3	1	2
7	3	1	3	2
8	3	2	1	3
9	3	3	2	1

In quality engineering, the Taguchi method is extremely effective to improve product quality while keeping the cost of design optimization low. In particular, experimental configurations are a systematic and efficient mechanism for exploring the domain of the design factors. It studies the effect of design factors simultaneously by planning matrix experiments using an orthogonal array, which studies a design factor domain with the smallest number of experiments [34]. For a product with 4 design factors with each design factor having 3 levels, a full factorial design requires 81 $\left(\text{i.e.: } 3^4\right)$ experiments. Only 9 experiments are sufficient to evaluate the main effect of each design factor in order to determine the optimum condition, when an orthogonal array, $L_9(3)$ (shown in Table 1), is used. Therefore 72 $\left(\text{i.e.: } 81-9\right)$ experiments are saved, compared with the full factorial design. In $L_9(3)$, there are four columns representing the design factors A, B, C and D, each of which has three levels. The number of rows represents the configurations of the product to be tested with respect to the experimental level defined by the row. The number of columns represents the maximum number of design factors which is studied, where the experimental levels defined by the columns are mutually orthogonal.

For example, in the first experiment, the four design factors have respective levels of one; in the second experiment, the four design factors have respective levels of either one or two. The first design factors are in level one and the last three design factors are in level two. Combinations in $L_9(3)$ have the pairwise balancing property, whereby every test setting of a design factor occurs with every experiment of all other design factors that have the same number of times. It minimizes the number of required experiments, while retaining the pairwise balancing property.

III. Short-term traffic flow predictor

The short-term traffic flow predictor conducts future traffic flow forecasting based on current and past traffic flow conditions, which are collected by the n detector stations (D_1, D_2, …, and D_n) as illustrated in Figure 1. D_1, D_2, …, and D_n are located between the on-ramp and the off-ramp of the freeway. D_i captures the average speed, $s_i(t)$, of vehicles passing by between time t and time $(t+T_s)$, where T_s is the sampling time. $s_i(t)$ reflects the traffic flow condition of the freeway at the location of D_i. If $s_i(t)$ is near the speed limit of the freeway, the traffic flow condition at the location of D_i is smooth. The output of the short-term traffic flow predictor indicates the prediction of the average speed of vehicles, $\hat{s}_L(t+m\cdot T_s)$, passing through the L-th detector station at time $(t+m\cdot T_s)$, where future traffic flow with m sampling time ahead is forecast. $\hat{s}_L(t+m\cdot T_s)$ as illustrated in Figure 2.

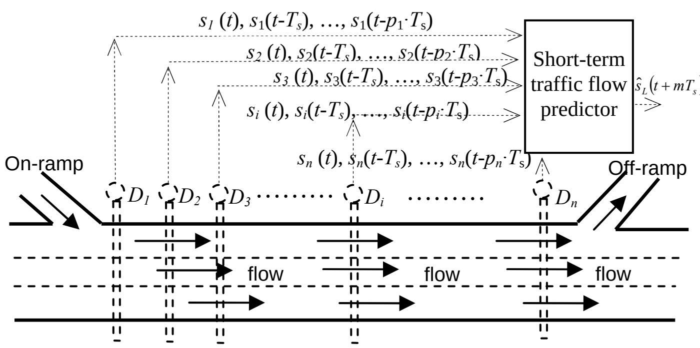


Fig. 1 Schematic of short-term traffic flow predictor for a section of the freeway

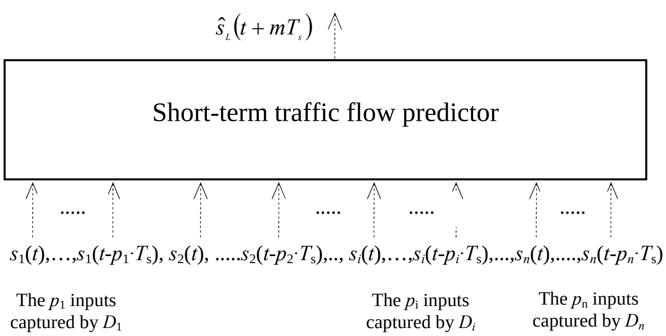


Fig. 2 Illustration of the input patterns and output of the short-term traffic flow predictor

$\hat{s}_L(t+mT_s)$ is formulated by a neural network with a fully connected cascade configuration, namely NN, as follows:

$$\hat{s}_L(t+m\cdot T_s)=\Psi\left(\sum_{i=1}^{n}\sum_{j=1}^{p_i}\gamma_{i,j}^{M}s_i(t-j\cdot T_s)+\sum_{k=1}^{M-1}w_k^{M-1}\Theta_k+\gamma_0^M\right), \tag{1}$$

where $\Theta_l=\Psi\left(\sum_{i=1}^{n}\sum_{j=1}^{p_i}\gamma_{i,j}^{M}s_i(t-j\cdot T_s)+\sum_{k=1}^{l-1}w_k^{l-1}\Theta_k+\gamma_0^{l-1}\right)$ with $M>l\geq 2$,

and $\Theta_1=\Psi\left(\sum_{i=1}^{n}\sum_{j=1}^{p_i}\gamma_{i,j}^{1}s_i(t-j\cdot T_s)+\gamma_0^1\right)$.

NN consists of parametrical factors and design factors. Three parametrical factors are represented by the NN weights which included: i) the weight on the connections between the k-th and the l-th hidden sets, w_k^l, with $l>k$; ii) the weights on the connections between the input sets and the k-th hidden sets, $\gamma_{i,j}^k$; and iii) the biases of the k-th hidden set, γ_0^k. Three design factors are included: i) the number of hidden nodes, M; ii) the activation function of the hidden set, $\Psi(.)$; and iii) the total number of input nodes, N_f, which is determined by $N_f=(p_1+p_2+...+p_n)$, where p_1, p_2,…, and p_n are the numbers of past traffic flow patterns collected from D_1, D_2, …, and D_n respectively.

The performance of NN can be evaluated by the mean absolute relative error (e_{MARE}), which indicates the differences between the true collected future traffic flow data and the predictions of NN. Based on the collected traffic flow data, e_{MARE} is formulated as equation (3):

$$e_{MARE}=\frac{1}{\|\mathbf{R}_{\mathbf{test}}\|}\sum_{[\theta(k),\varphi(k)]\in\mathbf{R}_{\mathbf{test}}}\frac{\left|\theta(k)-\hat{\theta}(\varphi(k))\right|}{\theta(k)}, \tag{3}$$

where $\hat{\theta}(\varphi(k))$ is the prediction of the NN, $\mathbf{R}_{\mathbf{test}}$ is the test data set with $[\theta(k),\varphi(k)]\in\mathbf{R}_{\mathbf{test}}$ and $k=1,2,...,\|\mathbf{R}_{\mathbf{test}}\|$. $\theta(k)$ is the average speed of vehicles collected from the L-th detection station at the time $(t(k)+m\cdot T_s)$; $\theta(k)$ is denoted by

$$\theta(k)=s_L(t(k)+m\cdot T_s); \tag{4}$$

as well as $\varphi(k)$ is the current and past traffic flow data, which is collected from the n detection stations and is denoted by:

$$\varphi(k)=\left[s_1(t(k)),s_1(t(k)-T_s),...,s_1(t(k)-p\cdot T_s),s_2(t(k)),s_2(t(k)-T_s),...,s_2(t(k)-p\cdot T_s),....s_n(t(k)-T_s),...,s_n(t(k)-p\cdot T_s)\right] \tag{5}$$

$s_j(t(k)-i\cdot T_s)$ is the average speed of cars collected at time $(t(k)-i\cdot T_s)$ by the j-th detection station, D_j, with j=1, 2, …,n, and $i=1,2,...,p\geq p_1,p_2,...,p_n$, where the number of input nodes of the NN is equal to, $p_1+p_2+...+p_n$. p is the number of pieces of past traffic flow data collected from all the detection stations

The determination of the optimal NN involves two main tasks: pre-defining the design factors and optimizing the parametrical factors. When the design factors are pre-defined, the optimization of the parametrical factors can be carried. The literature indicates that much research has been conducted on the optimization of parametrical factors of NN. For example, the back-propagation algorithm is a commonly used method to train NNs for short-term traffic flow forecasting from the past [4, 9, 10] and recent research [15, 42].

However, the trial and error method is still usually used for pre-defining the design factors. The values of p_1, p_2,…, and p_n determine the number of input nodes of the NN which may significantly affect the NN in forecasting future traffic flow. To pre-define the design factors, a systematical and effective method, namely the Taguchi method [33, 34], is proposed. It has been previously widely used to reduce variation in the quality characteristics of products and improve manufacturing robustness. The operations of the Taguchi method for designing short-term traffic flow predictors are detailed in Section 4.

IV. Design of Short-Term Traffic Flow Predictors Using Taguchi Method

In order to illustrate the use of the Taguchi method for designing short-term traffic flow predictors, a case study was conducted based on a real configuration of detection stations installed along a section of the Mitchell Freeway, Western Australia. Three detection stations, D_1, D_2 and D_3, are located at the intersection of Reid Highway and Mitchell Freeway: D_1 is located at the off-ramp of Reid Highway; D_2 is located between the off-ramp and on-ramp of Reid Highway; as well as D_3 is located at the off-ramp of Reid Highway. Four detection stations, D_4, D_5, D_6 and D_7, are located at the intersection of Hutton Street and Mitchell Freeway: D_4 is located at the off-ramp of Hutton Street; D_5 and D_6 are located between the off-ramp and on-ramp of Hutton Street; also D_7 is located at the off-ramp of Hutton Street. The distance between Reid Highway and Hutton Street is about 7 kilometres. The short-term traffic flow predictor is developed to forecast future traffic flow conditions with 2 sampling times ahead.

The traffic flow data sets were collected from the sixth week of 2009. They were collected over the 2-hour peak traffic period (6.30 – 10.30 am) on the five business days of the week, Monday, Tuesday, Wednesday, Thursday and Friday. Sixty seconds (1 minute) of sampling time were used and a total of 600 observations were included in each set of traffic flow data. Each traffic flow data set was divided into two sub-sets. The first sub-set of traffic flow data, namely the training data, collected from Monday to Thursday (comprising 80% of all the observations or 480 observations), was used for training the neural network models. The second sub-set of traffic flow data, namely the test data, collected from Friday (comprising 20% of all the observations or 120 observations), was used to evaluate the generalization capability of the trained neural network models. Data collected on Friday was used as test data and data collected from Monday to Thursday was used as training data, because we can use only the past data (collected from Monday to Thursday) to train the short-term traffic flow predictors and use the future data (collected on Friday) to evaluate the generalization capabilities of the short-term traffic flow predictors. It is not possible to use future data to train the short-term traffic flow predictors.

In the short-term traffic flow predictor, a complex relationship exists between the past and current traffic flow conditions which are captured by the seven detection stations, as well as the forecasted future traffic flow condition. The main objective of the proposed short-term traffic flow predictor is to accurately and reliably forecast future traffic flow conditions. The Taguchi method involving the following steps is proposed to optimize the topology of the short-term traffic flow predictor:

(A) Identification of design factors of short-term traffic flow predictors
(B) Specification of object functions for short-term traffic flow predictors
(C) Trial design for developing short-term traffic flow predictors
(D) Analysis of accuracies and reliabilities achieved by short-term traffic flow predictors

A. *Identification of design factors*

In the design of short-term traffic flow predictors, the design factors under consideration and their alternative levels are shown in Table 2.

These design factors are mostly related to determining the optimal topology of the short-term traffic flow predictor. Design factor A and Design factor B are critical for developing a NN for time series forecasting [2]. Design factor C to Design I are regarding the input nodes of the short-term traffic flow predictor, which are more significant for the design of NN configuration than is the determination of the number of hidden nodes for time series forecasting [39]. They are described by:

Design factor A: The number of hidden nodes in the hidden layers is an important design factor, which determines the size of the NN for the short-term traffic flow predictor. The number of hidden nodes recommended by [36] is $\log_2(480) \approx 10$. The maximum number of hidden nodes of 50 is recommended by [6] to model a set of benchmark modeling problems. Therefore, Level 1, Level 2 and Level 3 are set between these two settings. 10 hidden nodes, 20 hidden nodes and 50 hidden nodes are set as Level 1, Level 2 and Level 3 respectively.

Design factor B: For the activation functions of the hidden set, $\Psi(.)$, Tansig, Logsig and Purelin functions are commonly used. Level 1, Level 2 and Level 3 are defined as Tansig, Logsig and Purelin functions respectively.

Design factors C to Design factor I: the amounts of input patterns of the short-term traffic flow predictor, are determined by p_1, p_2, …, and p_7, with respect to the seven detection stations, D_1, D_2, …, and D_7 respectively. They represent the number of time sampling lags that are captured by the detection stations. 1 time sampling lags, 5 time sampling lags, and 10 time sampling lags are considered as Level 1, Level 2 and Level 3 respectively.

Table 2 Design factors of the short-term traffic flow predictors

Design factor		Level		
		1	**2**	**3**
Design factor A	**Number of hidden nodes**	10	20	50
Design factor B	**Activation function, $\Psi(.)$**	Tansig	Logsig	Purelin
Design factor C	**p_1 w.r.t D_1**	1	5	10
Design factor D	**p_2 w.r.t D_2**	1	5	10
Design factor E	**p_3 w.r.t D_3**	1	5	10
Design factor F	**p_4 w.r.t D_4**	1	5	10
Design factor G	**p_5 w.r.t D_5**	1	5	10
Design factor H	**p_6 w.r.t D_6**	1	5	10
Design factor I	**p_7 w.r.t D_7**	1	5	10

'w.r.t.' represents 'with respect to'.

B. *Specification of objective functions*

The performance measure represents two aspects of the short-term traffic flow predictor: the accuracy of traffic flow forecasting and the robustness against the noise factors. The objective function in terms of the signal-to-noise ratio (S/N) is shown in equation (6). It is intended to address these two aspects by defining a type of signal-target problem [25]. It evaluates the accuracy of the traffic flow predictor by comparing the outputs of the traffic flow predictor with the actual traffic flow conditions. It also evaluates the level of robustness against the noise factors. The ideal differences in terms of network accuracy should be zero. If the S/N is larger, then the error between the actual traffic flow conditions and the forecasting is smaller, as well as the robustness of accuracy is larger.

$$\eta = -10 \times \log_{10} \frac{1}{n-1} \sum_{i=1}^{n} \left(e_{MARE}^{i} - \mu \right)^2 \quad (6)$$

where η is the S/N ratio for the accuracy of the short-term traffic flow predictor; n is the number of trials with different initial values of NN weights; μ is the mean value of e_{MARE}^{i}; and e_{MARE}^{i} is defined by equation (3) which indicates the differences between the actual traffic flow conditions and the forecasts. Based on equation (6), noise factors, including the varieties of initial values of the NN weights between neural net nodes, can be addressed as external to the short-term traffic flow predictors.

C. Trial design

Within each design factor, there are three levels of interest. An orthogonal array $L_{27}(3^9)$ is used, because it has 3 levels and 9 design factors to match the requirements of the short-term traffic flow predictor. The orthogonal array $L_{27}(3^9)$ for this design problem is shown in Table 3. The elements at the intersections indicate the level settings that apply to the design factors for that combination of levels of a main trial. When the design factors (i.e. the NN configuration) are pre-defined, the parametrical factors of the short-term traffic flow predictor (i.e. the NN weights) can be determined. A recently developed learning algorithm, namely the Wilamowski's learning algorithm [46, 51], is used to determine the optimal NN weights with respect to the pre-defined NN configuration, because of its very good convergence.

Trials	A	B	C	D	E	F	G	H	I	Number of input patterns	e_{MARE}^{1}	e_{MARE}^{2}	e_{MARE}^{3}	e_{MARE}^{4}	$\frac{1}{4}\sum_{i=1}^{4} e_{MARE}^{i}$	Ranks of $\frac{1}{4}\sum_{i=1}^{4} e_{MARE}^{i}$	S/N
1	1	1	1	1	1	1	1	1	1	7	0.1416	0.152	0.1464	0.1423	0.1456	20	49.15
2	2	3	2	1	1	2	1	2	2	23	0.0771	0.0722	0.079	0.0764	0.0762	8	52.95
3	3	2	1	1	2	1	2	2	2	23	0.1309	0.1233	0.1218	0.116	0.123	14	47.76
4	3	3	1	2	3	2	1	1	2	28	0.0723	0.0727	0.0782	0.0715	0.0737	3	57.82
5	3	1	2	3	2	1	1	2	1	28	0.1532	0.1505	0.1485	0.1553	0.1518	27	51.65
6	1	2	3	2	1	1	2	1	2	28	0.1197	0.1207	0.1157	0.1158	0.1179	10	55.42
7	2	3	3	1	2	3	2	1	1	33	0.0752	0.0715	0.0811	0.076	0.076	7	52.38
8	2	1	1	2	1	2	2	2	3	32	0.1459	0.154	0.1534	0.1484	0.1504	24	52.83
9	3	2	2	1	3	2	3	1	1	33	0.1182	0.1161	0.1198	0.1234	0.1193	11	51.98
10	1	2	1	2	2	2	3	3	1	37	0.1251	0.1262	0.1216	0.1253	0.1245	16	56.45
11	2	1	3	2	3	1	1	3	1	38	0.1498	0.1481	0.1461	0.1526	0.1492	23	56.75
12	1	3	2	3	1	1	3	1	3	38	0.0746	0.0731	0.077	0.0738	0.0746	4	58.21
13	2	2	1	3	2	3	1	1	3	38	0.121	0.1237	0.1153	0.1239	0.1209	12	56.45
14	3	3	2	2	1	3	2	3	1	37	0.0682	0.0716	0.072	0.0767	0.0721	1	55.71
15	2	1	2	2	2	3	3	1	2	41	0.147	0.1459	0.1473	0.1455	0.1464	21	58.74
16	3	2	3	1	1	3	1	3	3	43	0.1235	0.1179	0.1207	0.122	0.121	13	55.46
17	1	3	1	3	3	3	2	2	1	42	0.0826	0.0815	0.0753	0.0813	0.0802	9	57.47
18	1	1	2	1	2	2	2	3	3	41	0.1521	0.1454	0.1456	0.1432	0.1466	22	52.55
19	2	2	3	3	1	2	3	2	1	42	0.1238	0.1239	0.1222	0.1242	0.1235	15	58.78
20	2	3	1	1	3	1	3	3	3	43	0.0747	0.0689	0.0775	0.0729	0.0735	2	54.88
21	2	2	2	3	3	1	2	3	2	46	0.1236	0.1276	0.1226	0.1281	0.1254	17	53.44
22	3	3	3	2	2	1	3	2	3	46	0.0806	0.0751	0.0693	0.0756	0.0751	5	48.98
23	3	1	1	3	1	3	3	3	2	47	0.1534	0.1511	0.1531	0.1492	0.1517	26	50.87
24	1	2	2	2	3	3	1	2	3	46	0.1112	0.158	0.1545	0.1582	0.1455	19	34.12
25	1	3	3	3	2	2	1	3	2	46	0.0773	0.0732	0.0757	0.0758	0.0755	6	52.72
26	3	1	3	3	3	2	2	1	3	51	0.1424	0.1393	0.1454	0.1448	0.1429	18	55.84
27	1	1	3	1	3	3	3	2	2	51	0.1518	0.1451	0.1582	0.1471	0.1505	25	49.86

Table 3 Orthogonal array, $L_{27}(3^9)$, and experimental results

For each main trial corresponding to each row of the orthogonal array $L_{27}(3^9)$, 4 random trials (i.e. n=4 in equation (6)) are used to establish the initial NN weights prior to each learning session. The four trials with respect to the 27 main trials, e_{MARE}^{1} e_{MARE}^{2} e_{MARE}^{3} and e_{MARE}^{4}, are shown in Table 3**.** Thus, a total of 108 trials (27 main trials with 4 random trials) are conducted in order to assess the robustness of the network performance against initial NN weights prior to each learning session. Note that there are essentially 27 main trials to be carried out, in order to study the effects of each of the seven design factors. The other 81 (=108-27) trials are basically replications of the 27 main trials. This is necessary in order to obtain a more precise estimation of the trial error with different initial NN weights for training the NN weights in the traffic flow predictor.

If the full factorial design is used, 6561 ($=3^8$) main trials are required to be carried out, where eight design factors and three levels in each design factor is used for this traffic flow predictor design. Hence, there are a total of, 26244 ($=6561 \times 4$) trials, which need to be conducted, as four random trials are conducted in each main trial. In this design problem, approximately ten seconds are required for each trial. 26244 trials require 262440 seconds to be conducted. Hereby, 72.9 hours or 3.03 full days are required to design a short-term traffic flow predictor. When the orthogonal array, $L_{27}(3^9)$, is used for this design problem, only 108 trials, or 0.3 hours (=1080 seconds) are required for the design of the short-term traffic flow predictor, which is much less than the number required by full factorial design. Comparing this with the full factorial design, 26136 (=26244-108) trials, or 72.6 (=72.9-0.3) hours can be saved in the design of the short-term traffic flow predictor. Therefore, a significant amount of computational effort and time can be saved by using the Taguchi method. It demonstrated the effectiveness of using the Taguchi method in traffic flow predictor design.

Also the average *MARE* of the four trials with respect to the 27 main trials, $(1/4)\cdot\sum_{i=1}^{4} e_{MARE}^{i}$, the ranks of the average *MARE*, and the number of input patterns to the NN are shown in Table 3. It shows that the 14th trial with 37 input patterns can achieve the smallest average *MARE*. It is smaller than those achieved by the 26th and 27th trials, involved 51 input patterns, which is the largest number of input patterns used. It is also smaller than those achieved by the 1st trial, involving 7 input patterns, which is the smallest number of input patterns used. Also, in the 11th to the 13rd trials, 38 input patterns with different configurations are used, but different results in term of *MARE* are obtained. Therefore, these results clearly show that the determination of appropriate input patterns involved in NNs is important to traffic flow forecasting. These results show that one should not simply use the maximum number of input patterns, but rather use the optimal number of input patterns, which helps to specify the optimal configuration of input nodes.

D. Analysis of accuracies and reliabilities

After a detailed trial plan for the short-term traffic flow predictor design is developed, the results for the trials are conducted. Four results with respect to each trial were collected as illustrated by e_{MARE}^{1}, e_{MARE}^{2}, e_{MARE}^{3}, and e_{MARE}^{4} in Table 3. These results were the accuracies of the trained NNs as defined by equation (3). The signal to noise ratios (S/N), η, were computed by using the equation (6), for each row of the orthogonal array $L_{27}(3^9)$. The compiled results for all trials are shown in Table 3.

As the combinations of design factors of each trial are orthogonal, the main effect of each design factor can be separated out [3,25]. The main effects of each design factor at each of the three levels are calculated and shown in Table 4. The main effects shown in the response table are calculated by taking the average from Table 3 for a design factor at a given level. As an example, the design factor D is at level three in the trials of, 5, 12, 13, 17, 19, 21, 23, 25 and 26. The average of the corresponding traffic flow condition is 55.05, which is shown in the response table under the design factor D at level three. The sensitivity of each design factor is computed by taking the difference between the largest and smallest main effect for a given design factor. Table 4 shows that the design factor H shows the greatest sensitivity, which means that the one which has the largest effect on the short-term traffic flow predictor is realized by varying the design factor H, which is the number of time sampling lags of the detection station D_6. Similarly, the design factor C shows the least sensitivity to the short-term traffic flow predictor. The main effects of all design factors are also shown graphically in Figure 4. Graphing the main effects of design factors can provide more insight at a glance, and it clearly shows that design factor H has much greater sensitivity than do of the other design factors.

Design Factors	A	B	C	D	E	F	G	H	I
Level 1	51.77	53.14	53.74	51.89	54.38	52.92	51.90	55.11	54.48
Level 2	55.25	52.21	52.15	52.98	53.08	54.66	53.72	50.50	53.29
Level 3	52.90	54.57	54.03	55.05	52.47	52.34	54.31	54.32	52.15
Sensitivity	3.472	2.362	1.871	3.161	1.915	2.319	2.408	4.620	2.333

Table 4 Main effects of S/N of each design factor

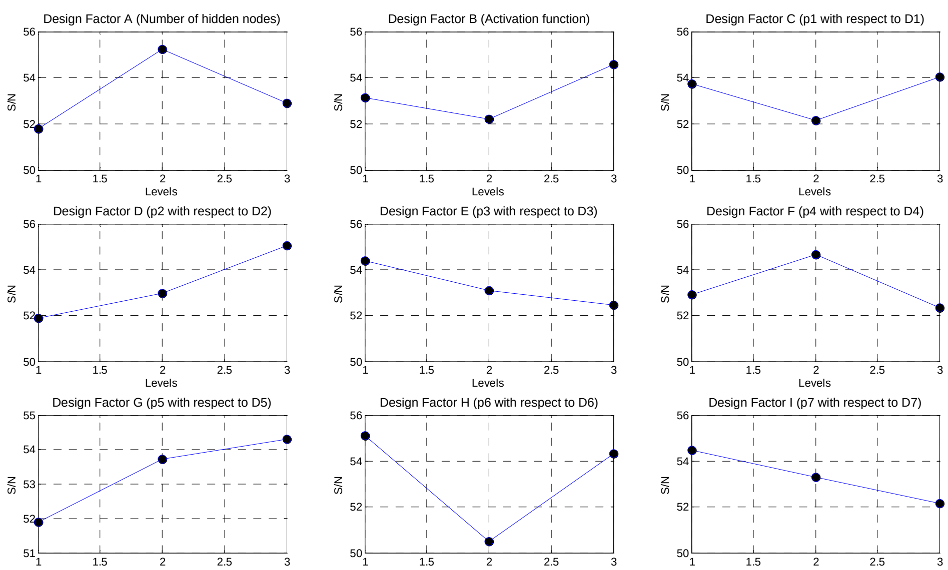


Figure 4 Main effects of each design factor

Based on Table 3 and Figure 4, the main effect of each level of each design factor can be observed. The design of short-term traffic flow predictors can be summed up as follows:

1. The sensitivities of the design factors C, and E, are smaller than those of the design factors A, B, D, F, G, H and I, where the design factors C and E represent the numbers of past traffic flow data, p_1 and p_3, collected from the detection

stations, D_1 and D_3 respectively. The design factors A, B, D, F, G, H and I, were considered to be more significant than the design factors C, and E, and these significant design factors were established for future designs. The insignificant design factors C, and E, could be kept unchanged for further designs. The insignificant design factors were relatively not carrying any variation of the outputs of the short-term traffic flow predictor.

2. The largest main effects of S/N of each design factor are underlined in Table 3, i.e. design factor A with level 2, design factor B with level 3, design factor C with level 3, design factor D with level 3, design factor E with level 1, design factor F with level 2, design factor G with level 3, design factor H with level 1, and design factor I with level 1. Figure 5 shows the simulation result obtained by the short-term traffic flow predictor, which was developed based on the design factor levels with smallest main effects. From the figure, it can be seen that the forecasted result is close to the actual traffic flow data.
3. Factors C to I specify different configurations of input patterns of the neural network for traffic flow forecasting. Different results in terms of forecasting accuracies can be obtained with different configurations. Therefore, an appropriate amount of input patterns is required to be specified, in order to obtain a satisfactory forecasting performance.

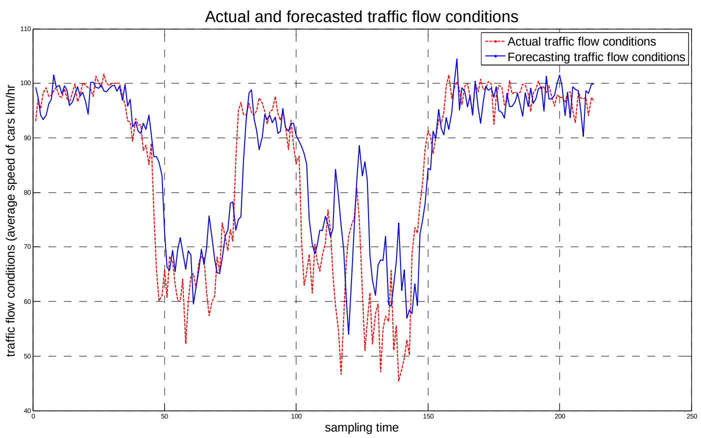


Figure 5 Forecasting of traffic flow condition

4. The optimum levels of the design factors can be refined further by decreasing the design factor ranges and increasing the number of levels of each design factor. As design factors A, D, and H are more significant than the rest of the design factors, further design modification can be carried out by refining the relatively significant design factors A, D, and H. However, for the purposes of the design of this short-term traffic flow predictor, an accuracy level higher than 94.51% can be obtained which is considered to be satisfactory.

V. Performance evaluation of the Taguchi method

This section demonstrates the effectiveness of the Taguchi method by comparing it with other existing methods, used to determine appropriate input node configurations for neural networks. Also, the effectiveness of the Taguchi method is further demonstrated by the determination of appropriate input node configuration for Type-II fuzzy neural networks, which have been applied to traffic flow forecasting.

A. Comparison with other existing methods

The two existing approaches, genetic algorithms (GA) [43] and particle swarm optimization (PSO) [45], which were developed for optimizing input node configurations of neural networks, were employed as a comparison. A fully connected cascade architecture was used. Both the chromosomes of the genetic algorithm and the particles of the particle swarm optimization algorithm are represented in two parts: the hierarchical string [44] and the integer string. The hierarchical string is used to represent the input node configurations of the neural network. As shown in Figure 6, the input node of the neural network is activated, when the corresponding element of the hierarchical string is '1'. When an element of the hierarchical string is '0', the corresponding input node of the neural network is not activated. The total number of '1's in the hierarchical string represents the number of activated input nodes. As mentioned previously in Section IV.A, there are seven detection stations, D_1, D_2, D_3, D_4, D_5, D_6 and D_7, which are used for collecting the current and past traffic flow data. Each of the detection stations captures 10 pieces of time series patterns into the neural networks for traffic forecasting, where each piece of time series patterns is inputted into its corresponding input node. Therefore, the total length of the hierarchical string in term of the input patterns is 70. The integer string is used to represent the number of hidden nodes used in the neural network, and the activation function, where the element for the number of hidden nodes is an integer ranging from 10 to 50, and the element for the activation functions represents either the 'Tansig', 'Logsig' or 'Purelin' function.

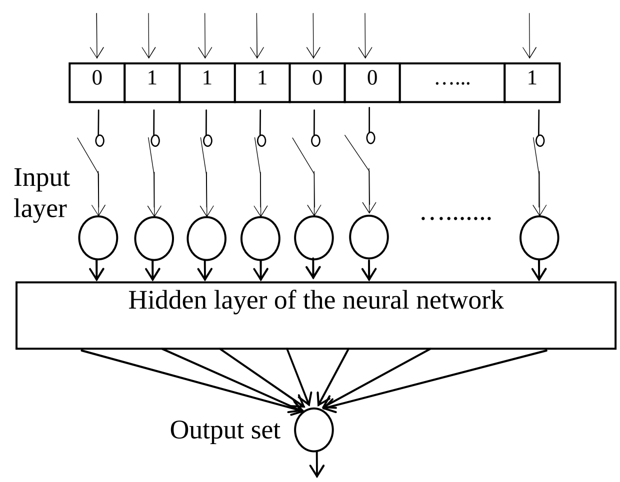


Figure 6 The hierarchical string in term of the input patterns for GA and PSO

In the GA [43], a population of chromosomes is first created randomly based on a hierarchical string illustrated in Figure 6. Then, each chromosome is evaluated based on equation (3) with respect to the input nodes specified by the chromosome, where the neural network weights are determined using the Wilamowski's learning algorithm [46]. After all evaluations, the genetic operations, crossover and mutation used in [43], are performed in order to reproduce new chromosomes to replace the old chromosomes, where the new chromosomes have higher potential to reach the optimal input node configuration than those of the old chromosomes. This GA process repeats until the pre-defined number of generations is reached. The detailed operations of the GA can be referenced in [43].

Here, two GAs, namely GA-4-27 and GA-20-50, were used. The following GA parameters were used in both GA-4-27 and GA-20-50: crossover rate =0.8; and mutation rate = 0.1. In GA-4-27, the relatively smaller population size with 4 chromosomes was used, and the pre-defined number of generations was set at 27. Hence, there were a total 108 of computational evaluations for each run, which was the same as the computational evaluations used in the Taguchi method. This setting was established in order to investigate any difference in performance between the GAs and the Taguchi method, when the same amount of computational effort was used in the two methods. In GA-20-50, the population size with 20 chromosomes was used and the pre-defined number of generations was set to 50. Hence, there were 1000 computational evaluations used for each run, where the number of computational evaluations used in the GA-20-50 was more than that used in the Taguchi method. We established this setting, in order to investigate whether GA-20-50 can achieve significantly better performance than the Taguchi method, when more computational effort is involved.

In the PSO approach, a swarm of particles is generated randomly by a discrete binary representation [45], which is identical to the one illustrated in Figure 6. Then, similar to the chromosome evaluations of the GA, each particle is evaluated based on equation (3) with respect to the input nodes specified by the particle, where the neural network weights are determined using the Wilamowski's learning algorithm [46]. After that, the positions and velocities of the particles in the swarm are improved based on their own best positions and their global best position found so far. The improvement process continues, until the pre-defined number of generations has been reached. The detailed operations of the PSO can be referenced in [45].

Here, two PSO, namely PSO-4-27 and PSO-20-50, were used. The following PSO parameters were used in both PSO-4-27 and PSO-20-50: the maximum and minimum inertia weights are set to 0.9 and 0.2, respectively; the initial acceleration coefficients are set to 2.0. In PSO-4-27, the swarm size with 4 particles was used and the pre-defined number of generations was set to 27. Hence, there were 108 computational evaluations for each run, which was the same as in the Taguchi method. By doing this, we can evaluate the performance of the PSO and the Taguchi method when the same amount of computational effort is involved. In PSO-20-50, the swarm size with 20 particles was used and the pre-defined number of generations was set to 50. Hence, there were 1000 computational evaluations for each run, which were more than in the Taguchi method. This allows us to determine whether the PSO can outperform the Taguchi method, when more computational efforts is involved.

As all the tested algorithms, GA-4-27, GA-20-50, PSO-4-27 and PSO-20-50, are stochastic algorithms, different results could be found with different runs. Therefore, GA-4-27, GA-20-50, PSO-4-27 and PSO-20-50, were run for 30 times, and the results of the 30 runs were recorded. Results in terms of solution qualities obtained by all methods and the computational times used for all methods are shown in Figure 7 and Figure 8, respectively. Figure 7 shows that the results obtained by both GA-4-27 and PSO-4-27 are poorer than those obtained by the Taguchi method, where the computational efforts used in the three methods, GA-4-27 and PSO-4-27 and the Taguchi method, were the same, as shown in Figure 8. Figure 7 shows that the results obtained by the Taguchi method are still slightly better than the ones obtained by the PSO-20-50, which is better than the GA-20-50. However, the computational efforts involved by both PSO-20-50 and GA-20-50 were significantly greater than those used by the Taguchi method.

These results indicate that the Taguchi method can obtain better results than those obtained by both the GA and the PSO, while the same computational efforts were involved in the three methods. Both GA and PSO can achieve solution qualities similar to those of the Taguchi method only when a significant amount of extra computational efforts was used. Therefore, the Taguchi method is more effective than both the GA and the PSO in searching for the appropriate input node configuration of the neural network for traffic flow forecasting.

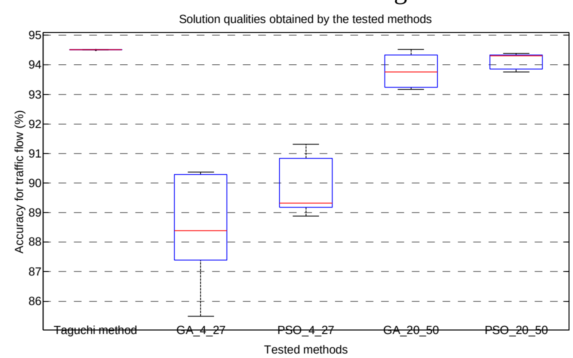


Figure 7 The quality of solutions obtained by the tested methods

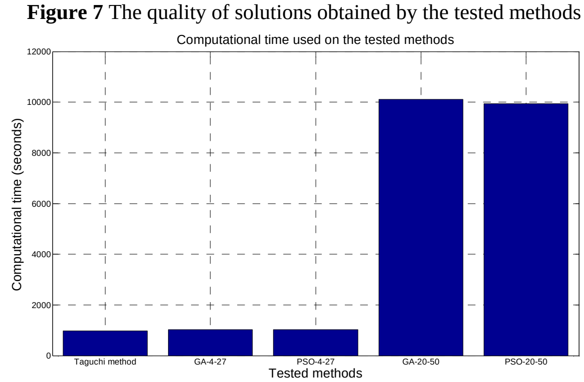


Figure 8 Computational time used by the tested methods

B. Type-II Fuzzy Neural Network Implementation

The effectiveness of the Taguchi method is further demonstrated by the determination of input node configurations of a Type-II fuzzy neural network (T-II-FNN) used or future traffic flow forecasting [47]. As using the data captured by all detection stations may not be most useful for the T-II-FNN in forecasting future traffic flow, it is essential to select the significant detection stations as the data sources of the input nodes. Figure 9 illustrates an input node configuration of the T-II-FNN with respect to the seven detection stations, where $s_1(t)$ is the traffic flow data captured by the detection station D_1; $s_2(t)$ is the traffic flow data captured by the detection station D_2, and so on. It shows that that D_1, D_3, D_6 and D_7 are used as input nodes of the T-II-FNN for traffic flow forecasting, while D_2, D_4, and D_5 are not used. The detailed structure of the T-II-FNN can be referred to [47].

Here, the Taguchi method is used to determine the appropriate input node configuration of the T-II-FNN. As there are seven detection stations, which are either ‘connected’ to or ‘disconnected’ from the T-II-FNN, an orthogonal array ($L_8\left(2^7\right)$) is used here for the design of T-II-FNN, where $L_8\left(2^7\right)$ is used for system design with seven design factors and two levels. $L_8\left(2^7\right)$, shown in Table 5, is used to conduct eight trials of input node configurations. For the first trial, all detection stations are disconnected from the T-II-FNN, so the forecasting accuracy is 0%, which is shown in the second row of Table 5. For the rest of the trials, four input nodes are connected to the T-II-FNN in order to forecast the future traffic flow. For the second trial, the detection stations, D_1, D_3, D_5 and D_7, are connected to the T-II-FNN and D_2, D_4, and D_6, are not connected. Based on this input node configuration, the T-II-FNN was developed (with the Matlab fuzzy logic toolbox), and the forecasting accuracy was found to be 87.03%, which is shown in the last column of the third row of Table 5. The results indicate that even if the number of input patterns used on the T-II-FNN is the same, the forecasting accuracies obtained are different when different input pattern configurations are used.

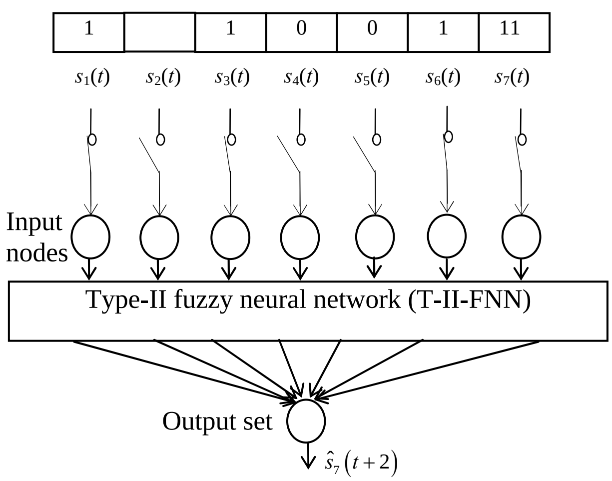


Figure 9 Input node configuration of the T-II-FNN

Trial	D_1	D_2	D_3	D_4	D_5	D_6	D_7	Forecasting accuracies
1st	0	0	0	0	0	0	0	0%
2nd	1	0	1	0	1	0	1	87.03%
3rd	0	1	1	0	0	1	1	79.05%
4th	1	1	0	0	1	1	0	85.56%
5th	0	0	0	1	1	1	1	80.78%
6th	1	0	1	1	0	1	0	87.06%
7th	0	1	1	1	1	0	0	82.39%
8th	1	1	0	1	0	0	1	84.99%

Table 5 The orthogonal array $L_8\left(2^7\right)$ and the trial results

After the eight trials have been conducted, the main effects of these detection stations can be calculated, and the most appropriate input node configuration is determined based on the calculated main effects. It has been determined that, when D_1, D_3, D_4 and D_6 are connected and D_3, D_5 and D_7 are disconnected, the largest forecasting accuracy (87.10%) is obtained.

If the full factorial design is used, 128 ($=2^7$) trials are required to be carried out, as there are seven design factors and each of them has two options either ‘connected’ or ‘disconnected’. Based on the 128 trials, the largest forecasting accuracy is found as 87.68%, when D_1, D_3, and D_4 are connected and D_3, D_5, D_6 and D_7 are disconnected. Both the forecasting accuracies obtained by the Taguchi method and the full-factorial design method, as well as both the computational time used by the two methods, are shown in Figure 10. These two figures clearly demonstrated that the forecasting accuracies obtained by the full-factorial design method are only slightly better than the one obtained by the Taguchi method. However, the computational time used by the full-factorial design method is 560.15 seconds, which is much longer than that required by the Taguchi method, which requires only 13.81 seconds. A significant amount of computational effort and time can be saved by using the Taguchi method. Based on this T-II-FNN design, the effectiveness of the Taguchi method can be once again demonstrated.

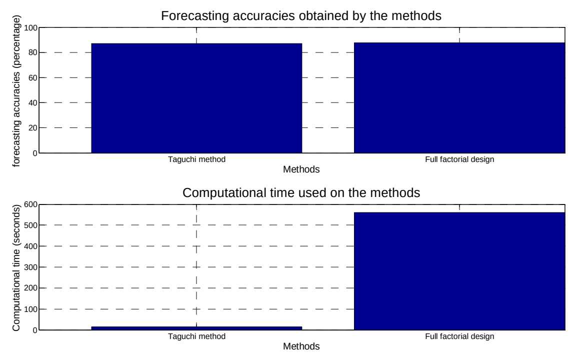


Figure 10 Forecasting performance obtained by the T-II-FNN

VI. Conclusions and Further works

In this paper, the Taguchi method, which is a robust optimization procedure for the design of high quality products or robust manufacturing processes, was proposed to be used for the topology design of neural network based short-term traffic flow predictors. The following four advantages were identified for the design of short-term traffic flow predictors, based on the Taguchi method:

1) It uses robustness as a significant design criterion of the short-term traffic flow predictor of which signal-to-noise ratio is used to evaluate the performance of the short-term traffic flow predictors. It intends to increase the solution quality in term of the accuracy of short-term traffic flow predictors.
2) It can address the relative importance of the design factors of short-term traffic flow predictors, where the design factors in terms of both the input node configurations and NN structures can be considered. This enables designers to evaluate the importance of the design factors concurrently, and to further refine the design factor ranges so as to achieve a better short-term traffic flow predictor.
3) It uses orthogonal arrays to systematically design a NN for short-term traffic flow forecasting. Thus, the design and development time for NNs can be reduced tremendously compared with the time required by the full factorial design and stochastic methods, such as genetic algorithms and particles swarm optimization.

4) It is not strictly confined to the design of NNs for short-term traffic flow forecasting. It has also been demonstrated that it can be used to develop short-term traffic flow predictors based on fuzzy neural networks such as T-II-FNN. Results show that the Taguchi method can assist in the rapid development of the best short-term traffic flow predictor to suit a particular traffic configuration.

This paper employs the Taguchi method to select appropriate input patterns of data captured by sensors, for the modeling and prediction of the traffic flow. The proposed method can be used in the following industrial applications:

a) Many industrial or manufacturing processes involve huge amount of sensor data for quality and operation control of new products. As not all captured sensor data is relevant for a specific purpose, the selection of useful sensor data is essential. Therefore, the Taguchi method presented in this paper can be applied very well in such industrial and manufacturing processes, in order to assist the quality and operation control for new products [48].

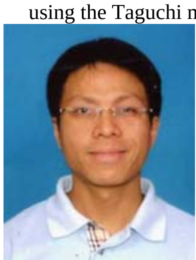
b) In order to identify and predict the lifetime of industrial cutting tools, data captured by sensors have to be used [50]. Not all data features captured by the sensors are helpful for the tool wear identification and prediction. The selection of significant features is important to reduce the effort in signal processing, as well as reduce the number of required sensors which will in turn reduce the costs. The Taguchi method is a very appropriate method to select significant features for such purposes.

c) For vehicle testing and diagnosis, huge amounts of data captured by sensors are required. Not all of the possibly captured data can be stored, because of the limited memory available in the tested vehicle. What is needed is an on-board preprocessing of data, in order to select useful data. The Taguchi method is a very good method for on-board preprocessing of data and can be successfully used for selecting useful data for vehicle testing and diagnosis purposes [49].

REFERENCES

1. B. Abdulhai, H. Porwal and W. Recker, "Short-term traffic flow prediction using neuro-genetic algorithms". ITS Journal 7, 3–41, 2002.
2. P.P. Balestrassi, E. Popova, A.P. Paiva, J.W. Marangon Lima, "Design of experiments on neural network's training for nonlinear time series forecasting", Neurocomputing, 72, 1160-1178, 2009.
3. D.M. Bryne and S. Taguchi, "The Taguchi approach to parameter design," ASQC Quality Congress, Anaheim, CA, p. 168-177, 1986.
4. S.C. Chang, R.S. Kim, S.J. Kim and B.H. Ahn, "Traffic flow forecasting using a 3-stage model", Proceedings of the IEEE Intelligent Vehicles Symposium, pp. 451-456, 2000.
5. K.Y. Chan, J. Singh, T.S. Dillon and E. Chang, "Traffic flow forecasting neural networks based on exponential smoothing method", Proceeding of the 6th IEEE Conference on Industrial Electronics and Applications, pp. 376 – 381, 2011.
6. B. Choi, J.H. Lee and D.H. Kim, "Solving local minima problem with large number of hidden nodes on two-layered feed-forward artificial neural networks", Neurocomputing, vol. 71, pp. 3640-3643, 2008.
7. G.A. Davis, and N.L. Nihan, "Nonparametric regression and short-term freeway traffic forecasting", Journal of Transportation Engineering, vol. 177, no. 2, pp. 178–188, 1991.
8. H. Dia, "An object-oriented neural network approach to short-term traffic forecasting", European Journal of Operational Research, vol. 131, pp. 253-261, 2001.
9. M. Dougherty, "A review of neural networks applied to transport", Transportation Research Part C: Emerging Technologies, vol. 3, no. 4, pp. 247-260, 1995.
10. M.S. Dougherty and M.R. Cobbett, "Short-term inter-urban traffic forecasts using neural networks", International Journal of Forecasting, vol. 13, pp. 21-31, 1997.
11. M.O. Efe, "Neural network assisted computationally simple PID control of a quadrotor UAV", IEEE Transactions on Industrial Informatics, vol. 7, no. 2, pp. 354-361, 2011.
12. Y. Gao and M.J. Er, "NARMAX time series model prediction: feedforward and recurrent fuzzy neural network approaches", Fuzzy Sets and Systems, vol. 150, pp. 331-350, 2005.
13. A. Gomperts, A. Ukil and F. Zurfluh, "Development and implementation of parameterized FPGA-based general purpose neural networks for online applications", IEEE Transactions on Industrial Informatics, vol. 7, no. 1, pp. 78-89, 2011.
14. B. Gunter, "A perspective on the Taguchi methods", Quality Progress, pp. 44-52, 1987.
15. S. Innamaa, "Effect of monitoring system structure on short-term prediction of highway travel time", Transportation Planning and Technology, vol. 29, no. 2, pp. 125-140, 2006.
16. C.F.J. Kuo, C.P. Tien and C.H. Chiu, "Analytical research on intellectual control of yarning characteristics for cotton collocation and rotor spinning", International Journal of Advanced Manufacturing Technology, vol. 32, pp. 764-773, 2007.
17. C.F.J. Kuo and Y.S. Wu, "Application of a Taguchi-based neural network prediction design of the film coating process for polymer blends", International Journal of Advanced Manufacturing Technology, vol. 27, pp. 455-461, 2006.
18. G. Lachtermacher and J.D. Fuller, "Backpropagation in time-series forecasting", Journal of Forecasting, vol. 14, pp. 381-93, 1995.
19. C. Ledoux, "An urban traffic flow model integrating neural network, Transportation Research", vol. 5C, pp. 287-300, 1997.
20. S.S. Mahapatra and A. Patnaik, "Study on mechanical and erosion wear behavior of hybrid composites using Taguchi experimental design", Materials and Design, vol. 30, pp. 2791-2801, 2009.
21. R.J. Mayer and P.C. Benjamin, "Using the Taguchi paradigm for manufacturing system design using simulation experiments", Industrial Engineering, vol. 22, no. 2 (1992) 195-209.
22. A. Ning, H.C.W. Lau, Z. Yi and T.T. Wong, "Fulfilment of retailer demand by using the mdl-optimal neural network prediction and decision policy", IEEE Transactions on Industrial Informatics, vol. 5, no. 4, pp. 495-506, 2009.
23. I. Okutani and Y.J. Stephanedes, "Dynamic prediction of traffic volume through Kalman filtering theory", Transportation Research, Part B: Methodology, vol. 18, no. 1, pp. 1–11, 1984.
24. M.S. Packianather, P.R. Drake and H. Rowlands, "Optimizing the parameters of multilayered feedforward neural networks through Taguchi design of experiments", Quality and Reliability Engineering International, vol. 16, pp. 461-473, 2000.
25. MS. Phadke, Quality Engineering Using Robust Design, (Prentice Hall, 1989).
26. C. Quek, M. Pasquier and B.B.S. Lim, "POP-TRAFFIC: A Novel Fuzzy Neural Approach to Road Traffic Analysis and Prediction", IEEE Transactions on Intelligent Transportation Systems, vol. 7, no. 2, pp. 133-146, 2006.
27. M. Sanjari, A.K. Taheri and M.R. Movahedi, "An optimization method for radial forging process using ANN and Taguchi method", International Journal Advanced Manufacturing Technology, vol. 40, pp. 776-784, 2009.
28. B.L. Smith, and M.J. Demetsky, "Traffic flow forecasting: comparison of modeling approaches", Journal of Transportation Engineering, vol. 123, no. 4, pp. 261–266, 1997.
29. B.L. Smith, B.M. Williams and R.K. Oswald, "Comparison of parametric and nonparametric models for traffic flow forecasting", Transportation Research Part C, vol. 19, pp. 303-321, 2002.
30. D. Srinivasan, C.W. Chan and P.G. Balaji, "Computational intelligence based congestion prediction for a dynamic urban street network", Neurocomputing, vol. 72, pp. 2710-2716, 2009.
31. A. Stathopoulos, L. Dimitriou and T. Tskeris, "Fuzzy modeling approach for combined forecasting of urban traffic flow", Computer-Aided Civil and Infrastructure Engineering, vol. 23, pp. 521-535, 2008.
32. L.P. Sullivan, "The power of Taguchi methods", Quality Progress, pp. 76-79, 1987.
33. G. Taguchi (1989) Introduction to quality engineering. Asian Productivity Organization, Tokyo.
34. G. Taguchi, Y. Yokohama, Y. Wu (1993) Design of experiments. ASI Press, Tokyo, Japan.

35. M.C. Tan, S.C. Wong, J.M. Xu, Z.R. Guan and P. Zhang, "An aggregation approach to short term traffic flow prediction", IEEE Transactions on Intelligent Transportation systems, vol. 10, no. 1, pp. 60-69, 2009.
36. N. Wanas, G. Auda, M. Kamel and F. Karray , "On the optimal number of hidden nodes in a neural network", IEEE 11th Canadian Conference on Electrical and Computer Engineering, vol 2, pp. 918-921, 1998.
37. Y. Wang, M. Papageorgiou and A. Messmer, "A real-time freeway network traffic surveillance tool", IEEE Transactions on Control Systems Technology, vol. 14, no. 1, 18-32, 2006.
38. H. Yin, S.C. Wong, J. Xu and C.K. Wong, "Urban traffic flow prediction using a fuzzy-neural approach", Transportation Research Part C, vol. 10, pp. 85-98, 2002.
39. G.P. Zhang, B.E. Patuwo and H.Y. Michael, "A simulation study of artificial neural networks for nonlinear time-series forecasting", Computers and Operations Research, vol. 28, pp. 381-396, 2001.
40. G. Zhang, B.E. Patuwo and M.Y. Hu, "Forecasting with artificial neural networks: the state of the art", International Journal of Forecasting, vol. 14, pp. 35-62, 1998.
41. Z. Zhao, Y. Zhang and H. Liao, "Design of ensemble neural network using the Akaike information criterion", Engineering Applications of Artificial Intelligence, vol. 21, pp. 1182-1188, 2008.
42. W. Zheng, D.H. Lee and Q. Shi, "Short-term freeway traffic flow prediction: Bayesian combined neural network approach", Journal of Transportation Engineering, vol. 132, no. 2, pp. 114-121, 2006.
43. F.H.F. Leung, H.K. Lam, S.H. Ling and P.K.S. Tam, "Tuning of the structure and parameters of neural network using an improved genetic algorithm", IEEE Transactions on Neural Networks, vol. 14, no. 1, pp. 79-88, 2003.
44. K.S. Tang, K.F. Man, Z.F. Liu and S. Kwong, Minimal fuzzy memberships and rules using hierarchical genetic algorithms, IEEE Transactions on Industrial Electronics, vol. 45, no. 1, pp. 162-169, 1998.
45. J. Yu, S. Wang and L. Xi, Evolving artificial neural networks using an improved PSO and DPSO, Neurocomputing, vol. 71, pp. 1054-1060, 2008.
46. B.M. Wilamowski, Neural network architectures and learning algorithms, IEEE Industrial Electronics magazine, vol. 3, no. 4, pp. 56-63, 2009.
47. L. Zhao, Short term traffic flow prediction based on interval type-2 fuzzy network networks, Communications in Computer and Information Science, pp. 230-237, 2010.
48. D. Wang, J. Liu and R. Srinivasan, data driven soft sensor approach for quality prediction in a refining process, IEEE Transactions on Industrial Informatics, vol. 6, no. 1, pp. 11-17, 2010.
49. H. Schweppe, A. Zimmermann and D. Grill, Flexible on-board stream processing for autormotive sensor data, IEEE Transactions on Industrial Informatics, vol. 6, no. 1, pp. 81-92, 2010.
50. J.H. Zhou, C.K. Pang, F.L. Lewis and Z.W. Zhong, Intelligent diagnosis and prognosis of tool wear using dominant feature identification, IEEE Transactions on Industrial Informatics, vol. 5, no. 4, pp. 454-464, 2009.
51. B. M. Wilamowski and H. Yu, "Improved Computation for Levenberg Marquardt Training," IEEE Transactions. on Neural Networks, vol. 21, no. 6, pp. 930-937, June 2010.
52. J.F.C. Khaw, B.S. Lim and L.E.N. Lim, Optimal design of neural networks using the Taguchi method, Neurocomputing, vol. 7, pp. 225-245, 1995.



Kit Yan Chan received PhD degree in Computing from London South Bank University, United Kingdom in 2006. Currently, he is a Senior Research Fellow at Curtin University, Australia. His research interests include computational intelligence and its applications to new product design, manufacturing process design, and traffic flow forecasting.

Saghar Khadem received the MSc degrees with emphasis on network and Distributed systems from Chalmers University of technology, Sweden in 2010. She is currently a PhD student in Curtin University, Australia. Her current research interests include artificial neural networks, traffic flow forecasting and control, and evolutionary computation.

Tharam Dillon is a Fellow of the IEEE, ACS and IE(Aust). He is the research professor at the Digital Ecosystems and Business Intelligence Institute, Curtin University, Australia. He is head of the IFIP International Task Force WG2.12/24 on Semantic Web and Web Semantics, the chairman of the IFIP WG12.9 on computational intelligence, the IEEE/IES Technical Committee on Industrial Informatics, and the IFIP Technical Committee 12 on Artificial Intelligence. He has published more than 750 papers published in international conferences and journals and is the author of 5 books and has another 5 edited books. His current research interests include Web semantics, ontologies, Internet computing, e-commerce, hy-brid neurosymbolic systems, neural nets, software engineering, database systems, and data mining.

Vasile Palade is currently working as a lecturer with the Department of Computer Science, University of Oxford, UK. He obtained his PhD at the University of Galati, Romania, in 1999. His research interests are in the area of computational intelligence with application to bioinformatics, fault diagnosis, web usage mining, among others. He published more than 80 papers in journals and conference proceedings as well as several books. He is an IEEE senior member.

Jaipal Singh received his PhD degree in computer science from La Trobe University, Melbourne, Australia in 2007. He has been doing research on network algorithms for more efficient routing, resource optimisation and improved quality of service. He is currently a research fellow at Curtin University working on short-term traffic forecasting, traffic visualisation and real-time congestion management. He is also developing new cyber-physical systems architectures in intelligent transportation systems.

Elizabeth Chang received her PhD in Computer Science from La Trobe University, Australia, in 1996. Since 2006, she is the professor at the Digital Ecosystems and Business Intelligence Institute, Curtin University, Australia. She has co-authored 3 books and has published over 350 scientific papers. Her research interest includes: ontology and multi-agent systems, data mining for business intelligence, trust, security and risk in e-Business, XML, Web Services, P2P for collaborative environments, web engineering, IT for business and commerce, IT for health informatics, and IT for education.