J. Mod. Transport. (2019) 27(4):306–316
https://doi.org/10.1007/s40534-018-0179-5

www.springer.com/40534

Traffic prediction using a self-adjusted evolutionary neural network

Shiva Rahimipour[1] · **Rayehe Moeinfar**[1] · **S. Mehdi Hashemi**[1]

Received: 1 July 2018 / Revised: 2 December 2018 / Accepted: 4 December 2018 / Published online: 22 December 2018
© The Author(s) 2018

Abstract Short-term prediction of traffic flow is one of the most essential elements of all proactive traffic control systems. The aim of this paper is to provide a model based on neural networks (NNs) for multi-step-ahead traffic prediction. NNs' dependency on parameter setting is the major challenge in using them as a predictor. Given the fact that the best combination of NN parameters results in the minimum error of predicted output, the main problem is NN optimization. So, it is viable to set the best combination of the parameters according to a specific traffic behavior. On the other hand, an automatic method—which is applicable in general cases—is strongly desired to set appropriate parameters for neural networks. This paper defines a self-adjusted NN using the non-dominated sorting genetic algorithm II (NSGA-II) as a multi-objective optimizer for short-term prediction. NSGA-II is used to optimize the number of neurons in the first and second layers of the NN, learning ratio and slope of the activation function. This model addresses the challenge of optimizing a multi-output NN in a self-adjusted way. Performance of the developed network is evaluated by application to both univariate and multivariate traffic flow data from an urban highway. Results are analyzed based on the performance measures, showing that the genetic algorithm tunes the NN as well without any manually pre-adjustment. The achieved prediction accuracy is calculated with multiple measures such as the root mean square error (RMSE), and the RMSE value is 10 and 12 in the best configuration of the proposed model for single and multi-step-ahead traffic flow prediction, respectively.

Keywords Traffic prediction · Neural networks · Genetic algorithm · Self-adjusted framework

1 Introduction

Intelligent transportation systems (ITSs) are expected to alleviate traffic problems around the world. Short-term traffic prediction is a highly researched area within ITS, and the results are used by transportation practitioners to reduce congestion and increase mobility. Efforts in this field started from the application of autoregressive integrated moving average (ARIMA) models and nonparametric techniques for traffic prediction. Since then, several parametric, nonparametric and also hybrid methods have been proposed by researchers. Basic parametric methods such as ARIMA [1], seasonal autoregressive integrated moving average method (SARIMA) [2] and Kalman filter [3] have been widely used in the literature. Developing these algorithms to meet the requirements of current engineering applications has been the subject of many research efforts in the past few decades [4]. For example, Luo et al. [5] proposed a hybrid prediction methodology based on improved SARIMA model and multi-input autoregressive (AR) model with genetic algorithm (GA) optimization, in order to provide a better prediction accuracy and also reduce the operation time.

Many nonparametric algorithms have also been proposed in this field. Huang and Sun [6] applied kernel regression with sparse metric learning to predict short-term traffic flow. In 2016, Habtemichael and Cetin [7] identified similar traffic patterns with an enhanced K-nearest

✉ Shiva Rahimipour
Rahimipour@aut.ac.ir

[1] Department of Mathematics and Computer Science, Amirkabir University of Technology (Tehran's Polytechnic), No. 424 Hafez Ave, Tehran 15875-4413, Iran

neighbor (KNN) algorithm and provided a data-driven short-term traffic prediction. Multilayer feedback NN [8] and KNN-based neuro-fuzzy system [9] are examples of applying NNs for short-term traffic prediction. Owing to their ability to approximate any degree of nonlinearity, NNs have been widely used in the literature. However, because of their developmental nature, a large degree of uncertainty is present when trying to select the optimal network parameters. To overcome this deficiency, researchers had to rely on very time-consuming and questionable rules of thumb [10].

An alternative approach for improving prediction accuracy is combining parametric, nonparametric and/or optimization algorithms to provide a hybrid method. In this approach, several methods are aggregated in order to provide a more efficient model. For instance, Hu et al. [11] used a combination of particle swarm optimization (PSO) and GA for traffic flow prediction. Cong et al. [12] proposed a model combined with support vector machine (SVM) and fruit fly optimization.

New interest in hybrid methods arises from the use of GAs. In 2015, Feng [13] analyzed the disadvantage of wavelet NNs and used GA to optimize the weight and threshold of NN. The GA has also been used to optimize NNs for different types of roads, to optimize links which connect input cells to hidden cells in the NN trained by Levenberg–Marquardt method [14] or to optimize the weights of the NN [15].

This study proposed a hybrid approach by applying GA optimization method to different kinds of NNs, such as simple back-propagation multilayer perceptron with "sigmoid" activation function and back-propagation multilayer feed-forward NN with momentum, to optimize network's architecture. The main difference of our NN structure from the existing ones is that we consider it as multi-output so that we can predict multi-step-ahead traffic flow with the original set of data. Main concerns of this study are as follows:

1. Multi-step-ahead prediction with NNs is usually provided with two approaches: (1) training separate NNs for each prediction horizon or (2) using one trained NN and sequentially predicting the traffic flow at time $t + 2$ using the predicted traffic flow at $t + 1$ and so on. The first approach is very time-consuming, and in the second approach, the accuracy of results decreases as the prediction horizon increases. The best approach is to use a multi-output NN which then raises the challenge of optimizing its parameters using a consistent optimization algorithm. In this paper, the result of applying a multi-objective optimization algorithm on multilayer perceptron (MLP) NNs is discussed.
2. The effort of this paper is to optimize these parameters for a multi-output MLP in a self-adjusted and evolutionary manner. Our goal is to reduce the dependency of final parameters on the manually initialized parameters.

The optimized model is used to predict multi-step-ahead flow at a given highway site considering both spatial and temporal features. Both temporal and spatial effects are essential for more accurate results.

The remainder of this paper is organized as follows: Sect. 2 presents the methodological and optimization framework used in this paper. Section 3 discusses the data used in this study and also the temporal and spatial representation of the traffic data. In Sect. 4 we present the empirical results, and finally, in Sect. 5 we discuss the findings of this paper.

2 Methodology

The framework employed for prediction entails two major blocks: the traffic estimator and its optimizer. The estimator structure is developed based on MLP NN with back-propagation learning algorithm. The optimizer used for setting the optimal set of variables is based on a specific kind of GAs called "non-dominated sorting genetic algorithm II" or briefly NSGA-II. This section gives a brief review on the properties of these two blocks.

2.1 MLP NNs

2.1.1 Standard back-propagation algorithm for MLP

The MLP belongs to the feed-forward NNs that are usually trained using the error back-propagation learning rule. The concept of a back-propagation MLP can be thought as a two-pass procedure through the different layers of the network: a forward one, in which the weights are fixed, and a backward pass, where the weights are adjusted according to the error correction rule. Consider the learning of a single neuron. Let us assume that a nonlinear activation function is chosen to be a hyperbolic tangent function, i.e.,

$$y_j = \varphi(u_j) = \tanh(\gamma_j u_j) = \frac{1 - e^{-2\gamma_j u_j}}{1 + e^{-2\gamma_j u_j}}, \quad (1)$$

where y_j is the actual output of neuron j, $\varphi(\cdot)$ is activation function, γ_j is the slope of the activation function, and $u_j = \sum_{i=1}^{n} w_{ji} x_i + \theta_j, \gamma_j > 0$, in which w_{ji} are synaptic weights from neuron j to i.

The aim of learning is to minimize the instantaneous squared error of the output signal by modifying the

synaptic weights, w_{ji}. This error can be calculated using Eq. (2).

$$E_j = \frac{1}{2}\left(d_j - y_j\right)^2 = \frac{1}{2}e_j^2, \tag{2}$$

where E_j is the instantaneous squared error of neuron j, d_j is the predicted output, y_j represents the actual output, and e_j is the difference between the predicted and actual output of neuron j.

The problem of learning can be formulated as follows. Given the current set of synaptic weights w_{ji}, we need to determine how to increase or decrease the local error function E_j. This can be done using the steepest descent gradient rule as follows:

$$\Delta w_{ij} = -\eta \frac{\partial E_j}{\partial w_{ji}}, \tag{3}$$

where η is a positive learning parameter determining the speed of convergence to the minimum [16].

This paper trains the MLPs with error back-propagation algorithm and two hidden layers with respect to four parameters: (1) learning parameter, (2) slope (gain) of the activation function, (3) number of the first layer's neurons, and (4) number of the second layer's neurons.

2.1.2 *Back-propagation algorithm with momentum updating*

The described learning algorithm has some important drawbacks. First of all, the learning parameter should be chosen small to provide minimization of the total error function E_j. However, for a small learning parameter, the learning process becomes very slow. On the other hand, while large values correspond to rapid learning, they lead to parasitic oscillations which prevent the algorithm from converging to the desired solution. Moreover, if the error function contains many local minima, the network might get trapped in some local minimum or get stuck on a very flat plateau. One simple way to improve the back-propagation learning algorithm is to smooth the weight changes by over-relaxation, i.e., by adding the momentum term (Eq. 4) [16].

$$\Delta w_{ji}^{[s]}(k) = \eta \delta_j^{[s]} O_i^{[s-1]} + \alpha \Delta w_{ji}^{[s]}(k-1), \tag{4}$$

where $\Delta w_{ji}^{[s]}(k)$ is the kth correction for weight w_{ji} in the sth layer, s is the layer number, O_i is the ith neuron output, $\delta_j^{[S]}$ is the local error in the sth layer, and $\alpha \in [0, 1]$ is the momentum. The second term is the so-called momentum term which may improve the convergence rate and the steady state performance of the algorithm (by damping parasitic oscillations).

More precisely, if we are moving through a plateau region of the performance surface function, then the gradient component $\frac{\delta E_j}{\delta W_{ji}}$ will be the same at each step and we can write

$$\Delta w_{ji} = -\eta \frac{\partial E_j}{\partial w_{ji}^{[s]}(k)} + \alpha \Delta w_{ji}^{[s]}(k-1) \cong -\frac{\eta}{1-\alpha} \frac{\partial E}{\partial w_{ji}^{[s]}(k)}. \tag{5}$$

This means that the effective learning rate increases to the value $\eta_{\text{eff}} = \frac{\eta}{1-\alpha}$ without magnifying the parasitic oscillations [16]. This NN is trained with respect to five parameters: (1) learning parameter, (2) slope (gain) of the activation function, (3) momentum term, (4) number of the first layer's neurons, and (5) number of the second layer's neurons.

2.2 NSGA-II

The main approach in multi-objective evolutionary algorithms (MOEAs) is to find a set of Pareto-optimal solutions in one single run. In multi-objective models, a set of Pareto-optimal solutions are reported instead of finding a single solution that optimizes all the objectives simultaneously.

In comparison with a number of MOEAs proposed in the past decade, NSGA-II is a well-known multi-objective GA proposed by Deb that finds a better spread of solutions in different problems [17].

This algorithm evaluates a set of solutions in a bi-/multi-directional search space, step by step. In each step, half of the solutions are picked as the elite set called parent population. In order to make a new set of solutions, genetic operators (crossover and mutation) are applied to the elite set to develop a child population. In the next step, members of the elite set are selected among the parent and child populations. Different versions of GAs move toward the optimum solution(s) based on this method and select the elite members with respect to fitness and spread. Several methods have been proposed for computing fitness and spread criteria. Fast non-dominated sorting is the method used in NSGA-II to arrange population in different ranks based on their fitness values. In this method, each solution is compared with every other solution in the population to find whether it is dominated. The spread criterion is measured by density estimation. Density estimation of solutions surrounding a particular point in the population is defined equal to the average distance of the two points on either side of this point along each of the objectives. The overall process of NSGA-II is shown in Fig. 1, where P_t is the adult population at time t, Q_t is the child population at time t, and R_t is the entire population.

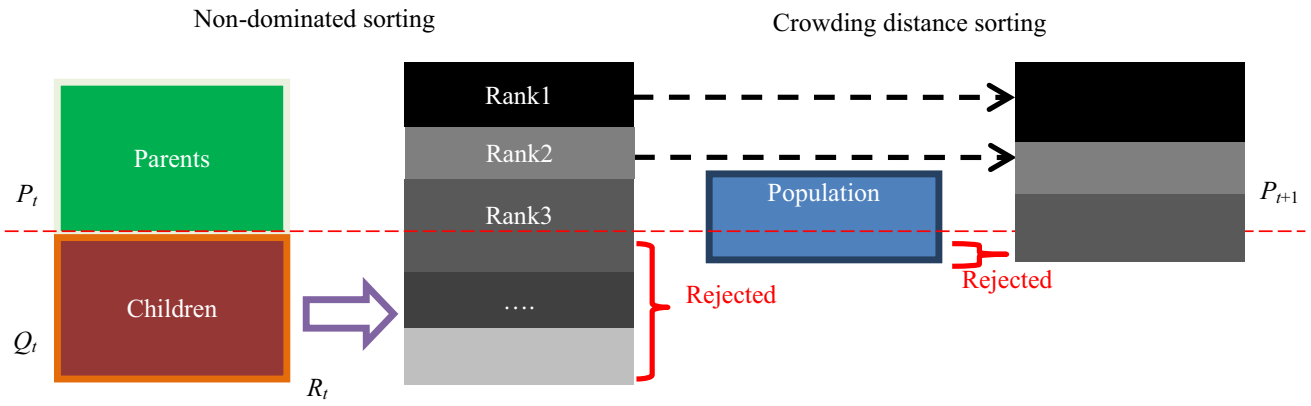

Fig. 1 NSGA-II procedure at generation time t [17]

3 Optimizing NN using NSGA-II

NNs' dependency on parameter setting is the major challenge in using them as a predictor. Given the fact that the best combination of NN parameters results in the minimum error of the predicted output, the main problem is NN optimization. So, it is viable to set the best combination of the parameters according to a specific traffic behavior. On the other hand, an automatic method—which is applicable in general cases—is strongly desired to address the appropriate NN's parameters. In this section, a self-adjusted framework is developed using an optimized NN for short-term prediction.

Most prediction systems are dependent on data transmission. This suggests that a continuous flow of data about traffic parameters is necessary to operate efficiently. However, it is common for most real-time traffic data collection systems to experience failures. So, a real-time prediction system must be able to generate predictions for multiple steps ahead to ensure its operation in cases of data collection failures [18].

One approach for multi-step-ahead predictions is to increase the number of outputs to achieve the single-/multi-step-ahead prediction at the same time. Consider $V(t)$ as the time series of traffic flow data which varies as a function of time. The input–output relation can be written as

$$[V(t), V(t+1)] = f(V(t-1), V(t-2), \ldots, V(t-n)), \tag{6}$$

where $f(\cdot)$ is the function that defines the relation between the historical traffic flows and target traffic flows (traffic flows at time t and $t+1$, in this case) and n is the look-back window size.

Although multi-step-ahead prediction is reckoned to be a proper solution in cases of failures of data collection systems, it was found in some previous relevant studies such as [10] that the correlation coefficient between actual and predicted flow series decreases as the prediction horizon increases. In order to solve this problem, we use a multi-output NN and used an MLP to optimize its parameters. The advantage of this combination is predicting multiple steps ahead through the original set of data with high accuracy (It will be shown that the correlation coefficient between actual and predicted flow series does not decrease as the prediction horizon increases). Optimizing the model using NSGA-II assures that we are getting the minimum error simultaneously for all steps ahead.

As previously discussed, the essential concern in modeling MLPs is the specification of their optimal parameters with respect to the number of hidden units and the learning rule. In a more microscopic view of optimizing these two issues, one can trace many network parameters such as learning rate, momentum rate, as well as the number of hidden units that have to be estimated by the practitioner. The most commonly used approach is to adopt a trial-and-error process in order to discover an optimal value for the three variable parameters [10]. The effort of this paper is to optimize these parameters for a multi-output MLP in a self-adjusted and evolutionary manner. The high-level flowchart presented in Fig. 2 shows the outline of our procedure. More detailed steps are provided in Fig. 3.

As illustrated in Fig. 3, the optimized value for the number of neurons in the first and second layers of the NN, learning ratio and slope of the activation function—shown by q1, q2, eta and gamma—is resulted from using NSGA-II to minimize NN error. In this process, EV1 and EV2 represent the error of validation data for one-step and two-step-ahead prediction, respectively. P and Q are the parents and children populations. The NNBP is the MLP utilized by back-propagation algorithm. After that, these efficient values are transferred to the NN algorithm to be set as the initial values of the mentioned parameters. The NN algorithm can be run once to provide the weights of links connected in three layers. These weights alongside the estimated parameters are structuring our final NN for multi-step-ahead prediction.

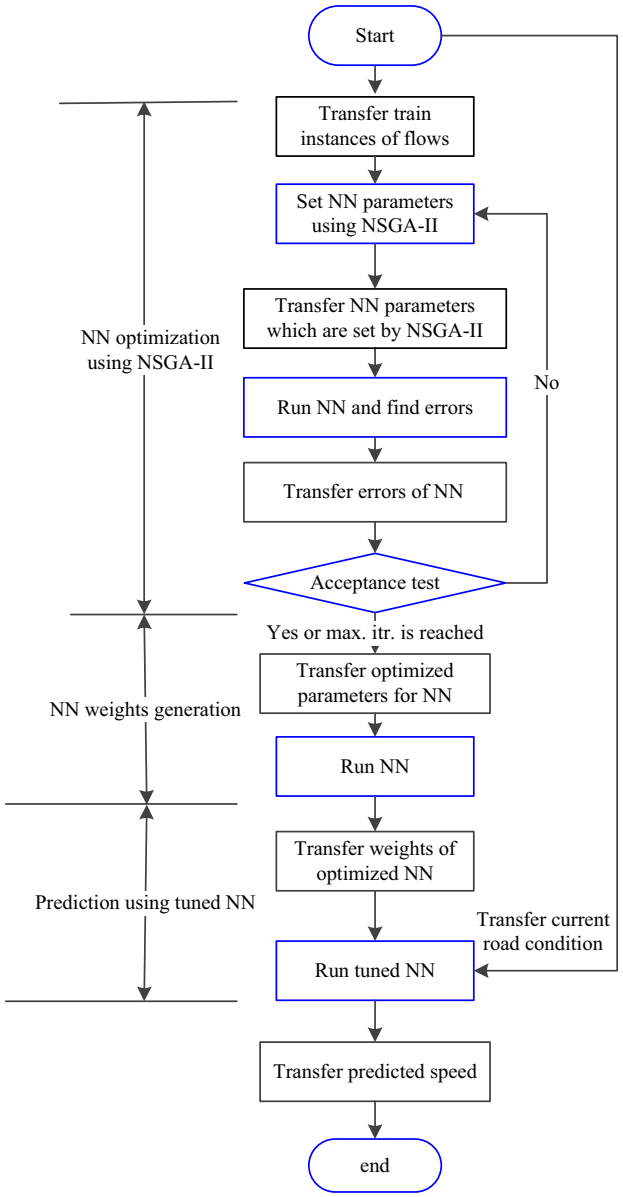

Fig. 2 Basic steps of optimizing multi-output MLP NN using NSGA-II

4 Study data

As previously mentioned in Sect. 3, the proposed framework is self-adjusted and data-independent. So, it is expected to predict the traffic flow for each set of data regardless of the different situations. In order to test the framework with a comprehensive set of data, data were collected from a six-lane section along the Hashemi Rafsanjani Highway/Tehran with 1,500 m length for three consecutive typical days. Traffic behavior during 24 h on this highway includes both congested and non-congested hours. Graphical representation of the selected area and organization of the loop detectors are shown in Fig. 4.

The traffic flow data are collected from certain data collection points every 5 min. In this paper, data are normalized to values between 0 and 1. The data samples used for training, validation and testing of the NNs are normalized using the min–max normalization technique given as follows:

$$X_n = \frac{X - X_{\min}}{X_{\max} - X_{\min}}, \tag{7}$$

where X_n is the normalized value, and $X_{\min}$ and $X_{\max}$ are an instance of the minimum and maximum values of the vector to be normalized. This reduces the possibility of reaching the saturation regions of the sigmoid transfer function during training. Figure 5 shows the traffic flow data in the selected area, in 5-min resolution.

4.1 Temporal and spatial representation of data

In this section, different datasets used in this study are discussed. Finding a way of incorporating both temporal and spatial characteristics of traffic data time series is a worth mentioning issue. For example, consider the traffic flow series which varies as a function of time, $V(t)$. Suppose that traffic data are collected using two detectors, one located in the desired section and the other one at the upstream. We name the data series collected by these detectors as $V_{\text{down}}(t)$ and $V_{\text{up}}(t)$, respectively. In order to predict the value of V_{down} at a given time t, the network must be trained using pairs of input–output values, where the input values could be

1. Time-lagged events of V_{down}, such as $V_{\text{down}}(t-1), V_{\text{down}}(t-2), \ldots, V_{\text{down}}(t-n)$.
2. Time-lagged events of V_{down} plus spatial attributes. In this case, input values are time-lagged events of both V_{down} and V_{up}, such as $V_{\text{down}}(t-1), V_{\text{down}}(t-2), \ldots, V_{\text{down}}(t-n), V_{\text{up}}(t-1), V_{\text{up}}(t-2), \ldots, V_{\text{up}}(t-n)$.

The first approach is considered as a univariate nonlinear prediction model, and the second one is a multivariate model.

The chosen highway section has three loop detectors: One is for collecting data at the desired section, and other two are placed at the upstream sections. Suppose that A and B represent the traffic data of the upstream sections (collected by No. 02 and No. 03 detectors, respectively) and C is for the downstream and the desired section (collected by No. 01).

Using the following relationship between these time series, the model can easily carry out traffic data forecasts:

$$C(t) = f(A(t-1), B(t-1), C(t-1), C(t-2), \ldots, C(t-n)).$$

Obviously, when input values only contain temporal attributes, it does not contain A and B time series:

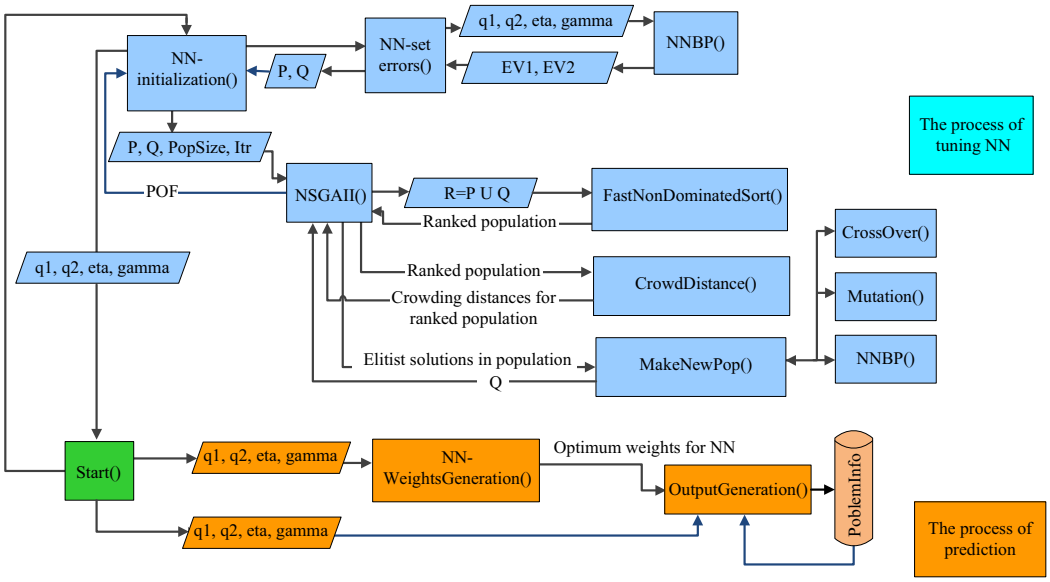

Fig. 3 The schematic representation of the optimization process

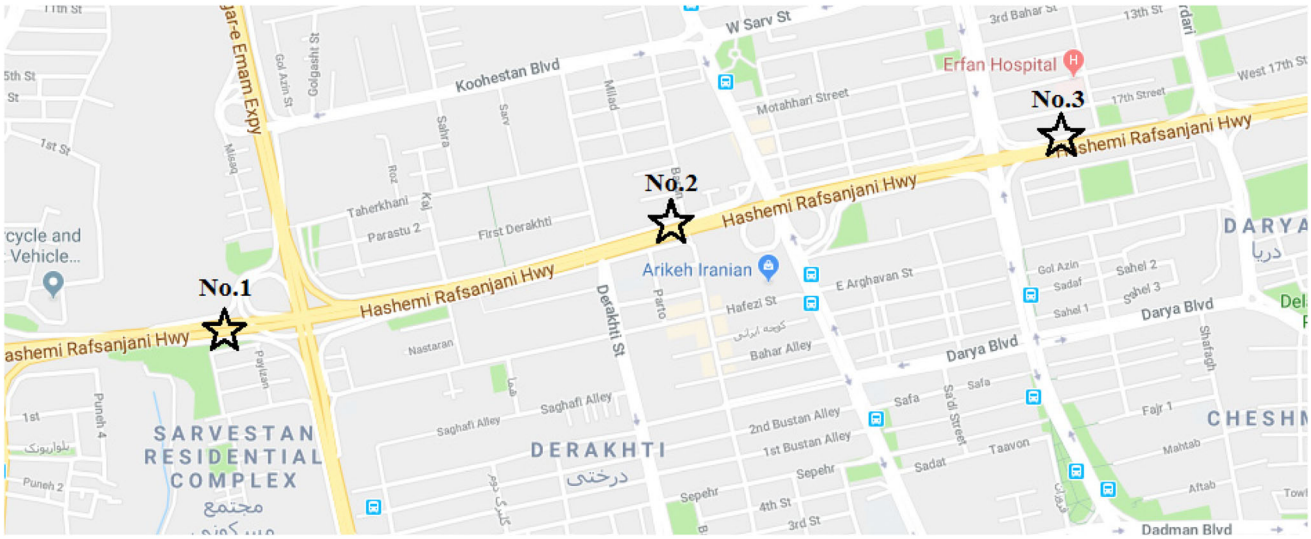

Fig. 4 Graphical representation of Hashemi Rafsanjani Highway

$$C(t) = f(C(t-1), C(t-2), \ldots, C(t-n)).$$

The main purpose of this paper is to discuss the optimization of NNs when we have more than one output. We also consider two input types containing temporal or spatiotemporal attributes. Here, network outputs will be one- and two-step-ahead predictions, which are the next 5 and 10 min. So simply assuming an MLP structure with two outputs, different types of input–output pairs used in this paper are as follows:

- Univariate (type 1 input):

$$[C(t), C(t+1)] = f(C(t-1), C(t-2), \ldots, C(t-n)).$$

- Multivariate (type 2 input):

$$[C(t), C(t+1)] = f(A(t-1), B(t-1), C(t-1), C(t-2), \ldots, C(t-n)).$$

4.2 Finding the input dimension

This section defines the aforementioned look-back window size or simply the input dimension. Increasing the input

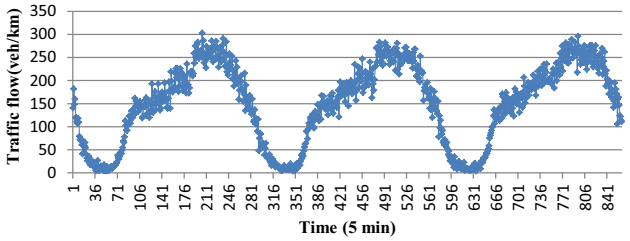

Fig. 5 Traffic flow behavior for 3 days

dimension of NNs can exponentially increase the computational complexity, but it may also increase the forecasting accuracy. Therefore, choosing the best dimension is a crucial issue. In this work, the statistical autocorrelation function (ACF) and partial autocorrelation function (PACF) are used for selecting the input dimension of a given time series in the nonparametric approach for traffic flow forecasting. Statistically, autocorrelation measures the degree of association between data in a time series separated by different time lags. The ACF is evaluated for various values of the lag time, and results are plotted. For traffic flows in 5-min intervals, the lag time will be in 5 min. Wherever the ACF curve intersects the lag time axis, its value is zero, indicating that $y(t - D)$ and $y(t)$ are linearly independent, where D denotes the look-back window size. The lag time corresponding to the first point of intersection is chosen as the optimum input dimension [19].

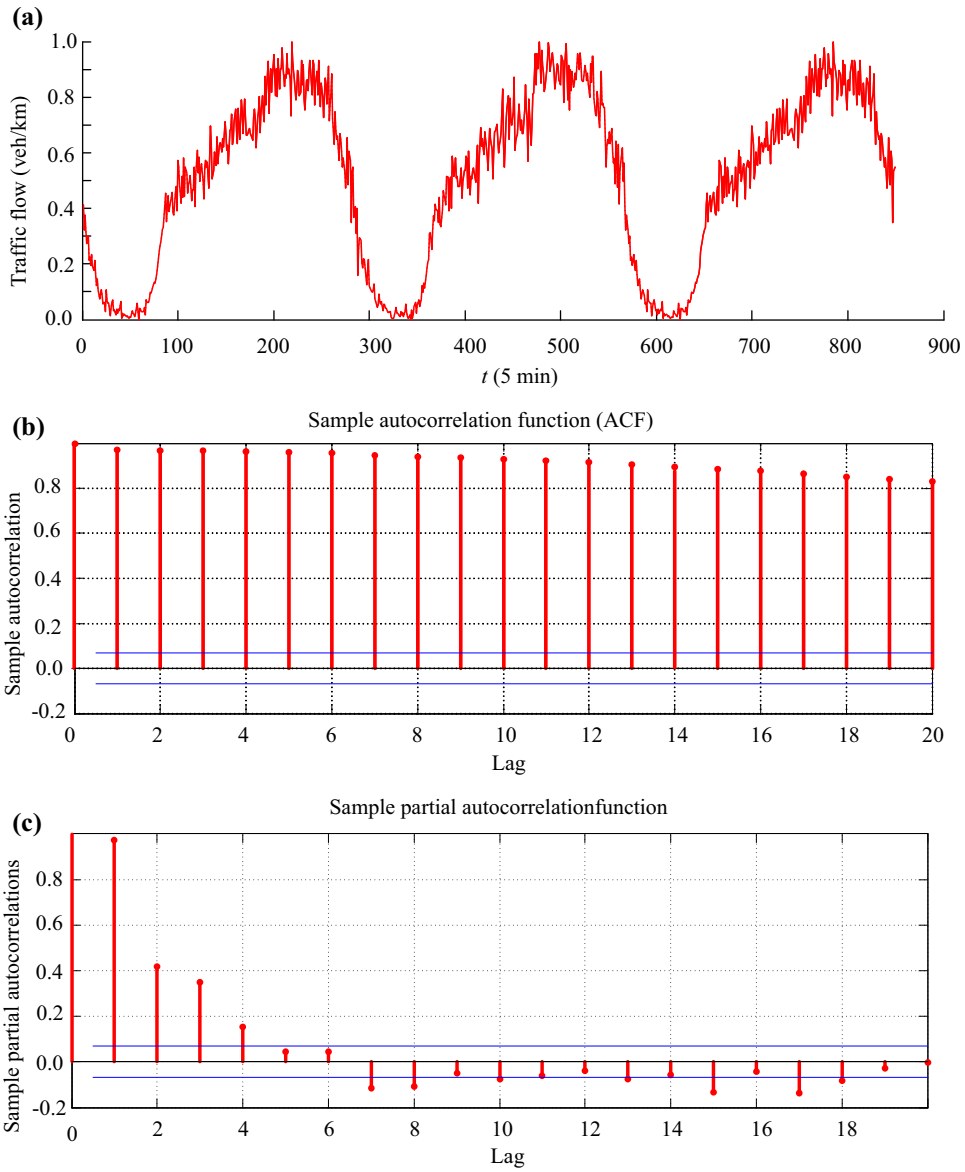

Fig. 6 Traffic flow measured in the selected two-lane freeway and its autocorrelation function: **a** time series plot, **b** autocorrelation histogram of the traffic flow data, and **c** partial autocorrelation histogram of the traffic flow data

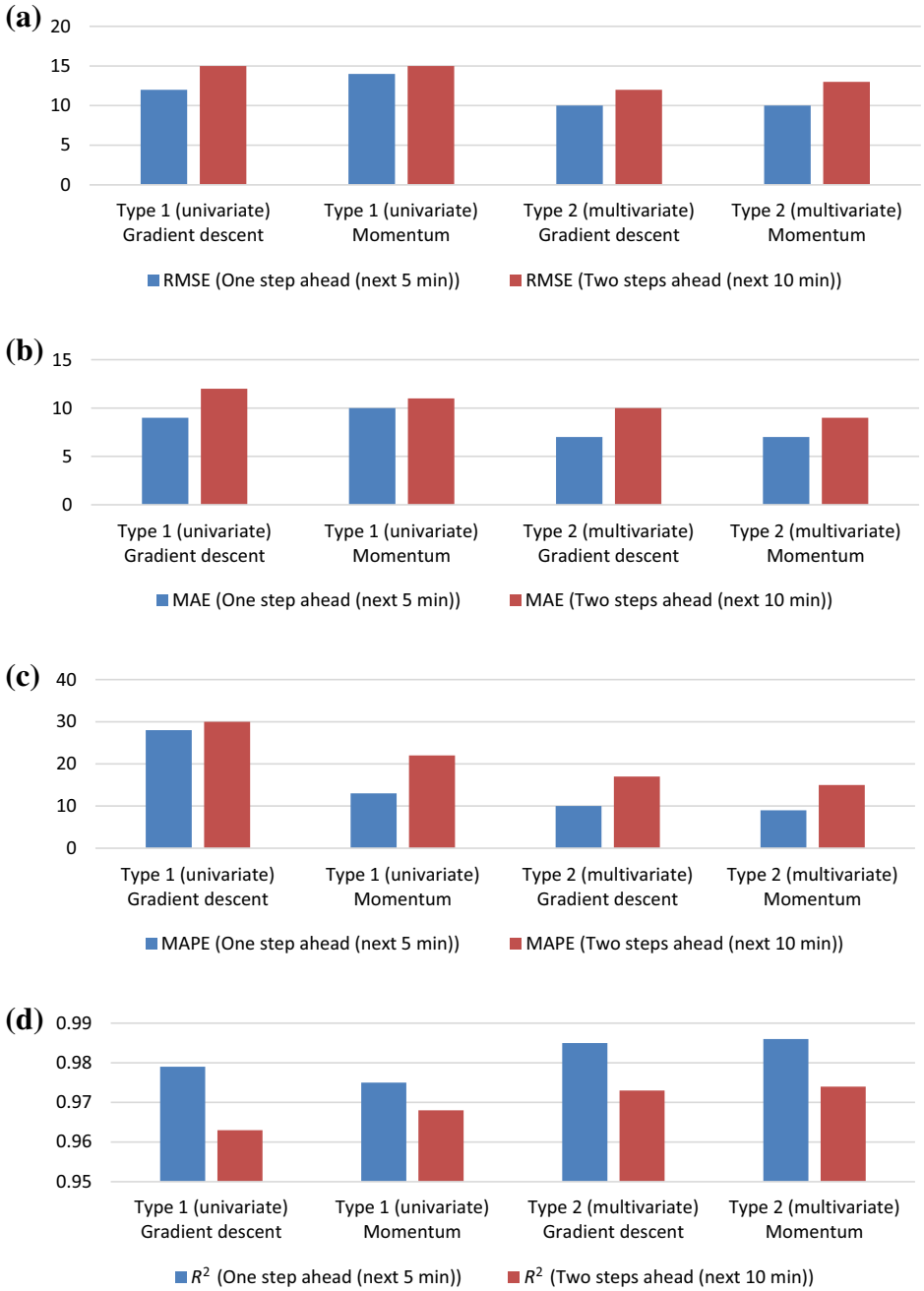

Fig. 7 Summary of the results: **a** RMSE values, **b** MAE values, **c** MAPE values, and **d** R^2 values for different data types (univariate vs. multivariate), different learning algorithms (back-propagation with/without momentum) and different prediction horizons (5 min vs. 10 min ahead)

Table 1 Comparison results

Forecasting 10 min ahead with type 2 input data	MAPE	R^2
Optimized NN with gradient descent	17	0.97
Seasonal ARIMA	18	0.92
Historical average	20	0.90

Figure 6a shows the time series plot of 850 and 5-min traffic flow data continuously recorded over a period of 3 days, that is, the number of observations $N = 850$. It displays a strong seasonal periodical pattern of 24 h (1 day), as expected. Figure 6b shows the variation of the ACF for the shown traffic flow data in Fig. 6a, with a lag time of up to 20 (5-min). The autocorrelation plot shows that the sample autocorrelations are very strong and positive and decay very slowly.

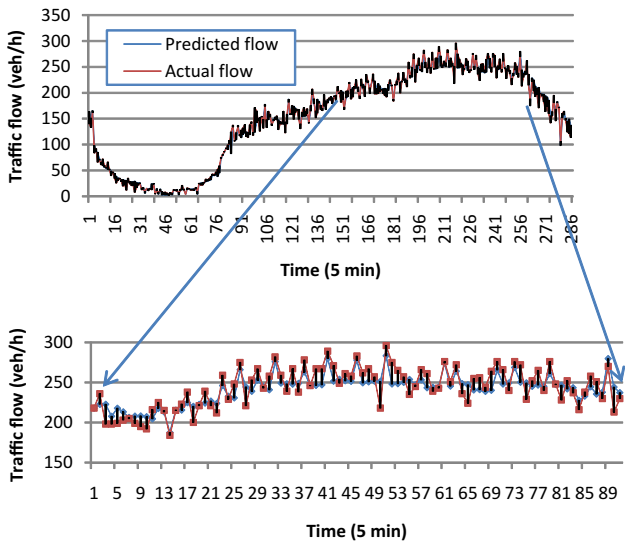

Fig. 8 Actual versus predicted flow of the univariate model for 5 min ahead

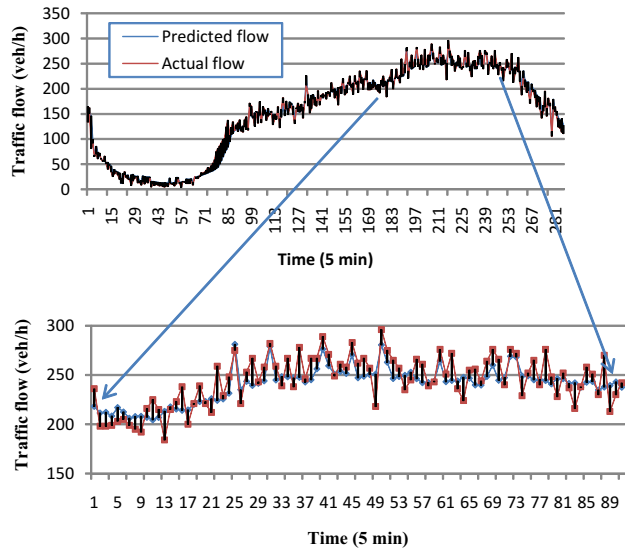

Fig. 9 Actual versus predicted flow of the univariate model for 10 min ahead

The next step is to produce the partial autocorrelation plot of the data. The partial autocorrelation plot of the data with 95% confidence bands shows that only the partial autocorrelations of the first, second, third and fourth lag are significant.

The PACF curve enters the confidence band at $D = 4$, indicating that $y(t-4)$ and $y(t)$ are linearly independent. The lag time $D = 4$ is therefore chosen as the optimum value to be used in the input dimension (Fig. 6c).

In other words, a four-dimensional input traffic flow vector, including four time-lagged periods of flow from No. 01 and two output units (representing traffic flow for No. 01 at $t+1$ and $t+2$ time intervals), will be used to model the univariate set of data, and a six-dimensional input traffic flow vector, including four time-lagged periods of flow from No. 01, one time-lagged periods of flow from both No. 02 and No. 03, and two output units (representing traffic flow for No. 01 at $t+1$ and $t+2$ time intervals), will be used to model the multivariate set of data.

5 Empirical results

In order to train and predict with NNs, 50% of the available data is used for training, 25% for validation, and the rest 25% for testing. Mean square error (MSE) for validation data has been used as a measure of the network's performance in the process of tuning NN. A set of elitist parameters are transferred to the next iteration; finally, the test data set is presented to the network to evaluate the performance of the adjusted network. Root mean square error (RMSE), mean average error (MAE) and the correlation coefficient known as R^2, between the actual and predicted flow series, are the performance measures used to compare the predicted values against the actual values. These measures are defined in Eqs. (8)–(10):

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(S_i - V_i)^2}{N}}, \tag{8}$$

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N}(S_i - V_i)}{N}, \tag{9}$$

$$\mathrm{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|S_i - V_i|}{V_i} \times 100, \tag{10}$$

where S_i is the predicted value for observation i, V_i is the actual value for observation i, and N is the number of observations.

Prediction results are summarized in plots presented in Fig. 7. From the figures, all of the errors (MAE, RMSE and MAPE) are sufficiently small to show that NSGA-II can be used as a promising tool for optimizing NN's parameters, with the aim of predicting the real-world traffic behavior.

Comparing errors in detail based on NN types, we find that using techniques like adding the momentum rate to the classic gradient descent method did not improve the performance. Both models have similar performance, and it is notable that the R^2 values in the models are very close and high for both one- and two-step-ahead predictions. Comparing errors based on data types (univariate and multivariate) shows that using spatial attributes in addition to the temporal ones improves the performance.

The aforementioned findings suggest that the multivariate approach can be used for traffic prediction at the selected highway site according to its predictive accuracy.

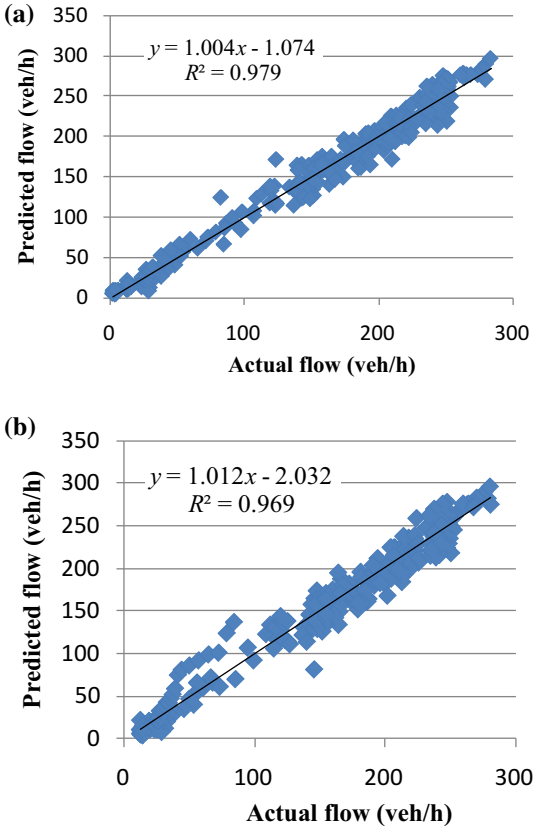

Fig. 10 Scatter plot of the univariate model: **a** 5 min ahead; **b** 10 min ahead

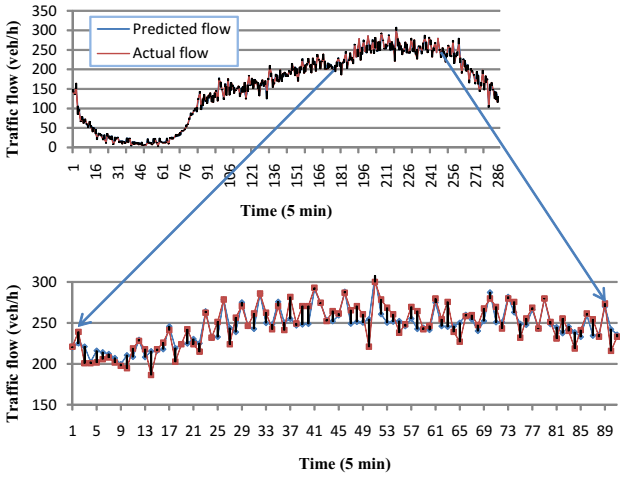

Fig. 11 Actual versus predicted flow of the multivariate model for 5 min ahead

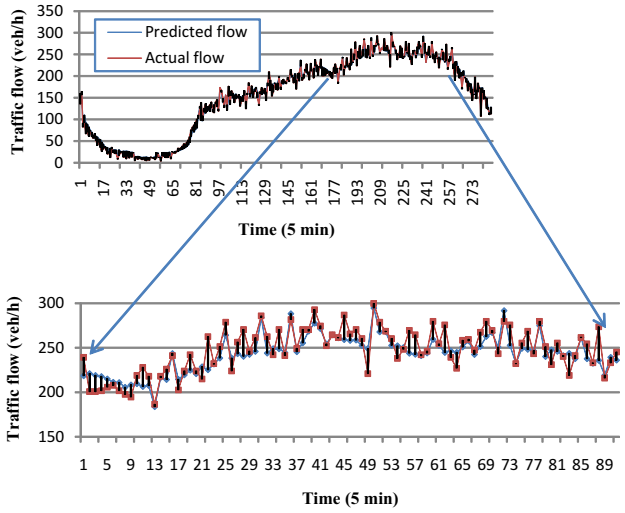

Fig. 12 Actual versus predicted flow of the multivariate model for 10 min ahead

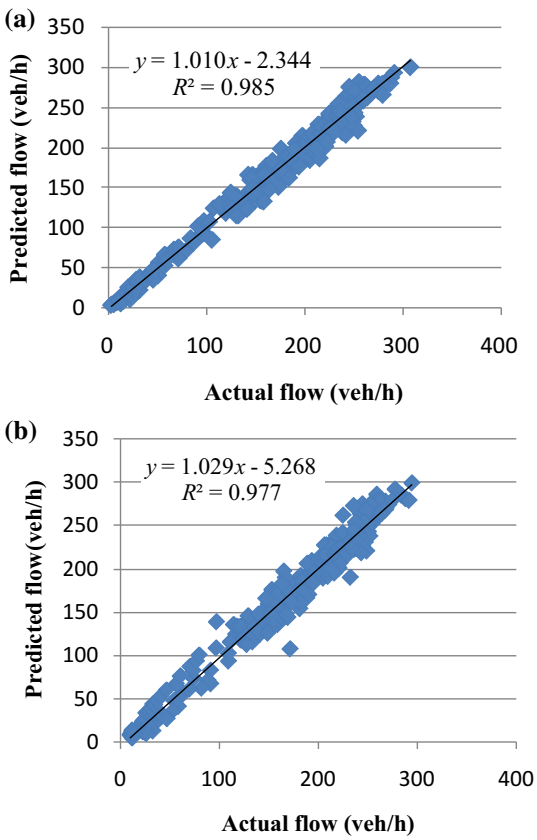

Fig. 13 Scatter plot of the multivariate model: **a** 5 min ahead; **b** 10 min ahead

In order to compare the accuracy of the proposed approach, multi-step-ahead prediction with the same data set is also provided with two other approaches: (1) seasonal ARIMA and (2) historical average. Table 1 shows the comparison results. MAPE and R^2 values are chosen to be reported for 10-min ahead prediction and with type 2 data as the models' input. Our observations revealed that both the seasonal ARIMA and the optimized NN can achieve good forecast in application to traffic flow, but the

developed predictive model using multi-output NN optimized with NSGA-II causes higher correlation between the predicted and actual traffic flows. This means that the traffic flow pattern predicted by the proposed approach is more coincident with the real traffic flow pattern. While the optimized NN results are comparable to the seasonal ARIMA, they both have better results than the simple historical average method.

Figures 8 and 9 show the predicted versus actual values of flow for the univariate model, for the next 5 and 10 min, respectively. Because of the similar results for both NNs, we only plot the results of genetically optimized gradient descent, for univariate and multivariate inputs. A portion of data points is magnified to clearly depict the differences. Figure 10 shows the scatter plot of the results shown in Figs. 8 and 9.

Similarly, Figs. 11 and 12 are for the multivariate approach and Fig. 13 shows the scatter plot.

6 Conclusion

The ability to predict the future values of traffic parameters helps to improve the performance of traffic control systems. Both single/multi-step-ahead predictions play a significant role in this field, but in cases of system failure, multi-step-ahead predicted values become beneficial. In order to avoid the low accuracy of long-term forecasts, instead of applying the iterated approach to the results of a single output model, multi-output NN is used in this study. This paper applied the NSGA-II to optimize the parameters of NNs with different learning rules and different types of inputs. This specific genetic algorithm is a well-known multi-objective genetic algorithm that finds a better spread of solutions in different problems.

The proposed framework predicts traffic flow values based on their recent temporal and spatial profiles at a given highway site during the past few minutes. Both temporal and spatial effects are found to be essential for more accurate prediction. Moreover, it was found that longer extent of prediction does not decrease the accuracy of results in this model. The model performance was validated using real traffic flow data obtained from the field.

This paper demonstrates the ability of this class of genetic algorithms to produce the best combination of network parameters. Results obtained from test data adduce that the model generalization ability is satisfactory. For the case of 5- and 10-min prediction horizon, R^2 indices are at least 0.98, which evidently shows the model generalization ability.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Voort MVD, Dougherty M, Watson S (1996) Combining Kohonen maps with ARIMA time series models to forecast traffic flow. Transp Res Part C 4(5):307–318
2. Williams BM, Hoel LA (2003) Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results. J Transp Eng 129(6):664–672
3. Gong Y, Zhang Y (2013) Research of short-term traffic volume prediction based on Kalman filtering. Paper presented at the 6th international conference on intelligent networks and intelligent systems, Shenyang, China
4. van Lint H, van Hinsbergen C (2012) Short-term traffic and travel time prediction models. In: Artificial intelligence applications to critical transportation issues. Transportation Research Circular, Number E-C168, pp 22–41
5. Luo X, Niu L, Zhang S (2018) An algorithm for traffic flow prediction based on improved SARIMA and GA. KSCE J Civ Eng 22(10):1–9
6. Huang R, Sun S (2013) Kernel regression with sparse metric learning. J Intell Fuzzy Syst 24(4):775–787
7. Habtemichael FG, Cetin M (2016) Short-term traffic flow rate forecasting based on identifying similar traffic patterns. Transp Res Part C 66:61–78
8. Hou Y, Edara P, Sun C (2015) Traffic flow forecasting for urban work zones. IEEE Trans Intell Transp Syst 16(4):1761–1770
9. Wei CC, Chen TT, Lee SJ (2013) KNN based neuro-fuzzy system for time series prediction. Paper presented at the 14th ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing, Honolulu, USA, pp 569–574
10. Vlahogianni EI, Matthew GK, Golias JC (2005) Optimized and meta-optimized neural networks for short-term traffic flow prediction: a genetic approach. Transp Res Part C 13:211–234
11. Hu W, Yan L, Liu K, Wang H (2016) A short-term traffic flow forecasting method based on the hybrid PSO-SVR. Neural Process Lett 43:155–172
12. Cong Y, Wang J, Li X (2016) Traffic flow forecasting by a least squares support vector machine with a fruit fly optimization algorithm. Procedia Eng 137:59–68
13. Feng G (2015) Network traffic prediction based on neural network. Presented at the international conference on intelligent transportation, big data and smart city (ICITBS), pp 527–530
14. Afandizadeh SH, Kianfar J (2009) A hybrid neuro-genetic approach to short-term traffic volume prediction. Int J Civ Eng 7(1):41–48
15. Cui J (2010) Traffic prediction based on improved neural network. J Converg Inf Technol (JCIT) 5(9):85
16. Cichoki A, Unbehauen R (1993) Neural networks for optimization and signal processing. Wiley, Chichester
17. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multi objective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6:182–197
18. Eleni IV, Matthew GK (2010) Urban transport and hybrid vehicles, local and global iterative algorithms for real-time short-term traffic flow prediction
19. Jiang X, Adeli H (2005) Dynamic wavelet neural network model for traffic flow forecasting. J Transp Eng 131(10):771