PATTERN RECOGNITION BASED SPEED FORECASTING METHODOLOGY FOR URBAN TRAFFIC NETWORK

Tamás TETTAMANTI[1*], Alfréd CSIKÓS[2], Krisztián Balázs KIS[3],
Zsolt János VIHAROS[4], István VARGA[5]

[1, 5]*Dept of Control for Transportation and Vehicle Systems,
Budapest University of Technology and Economics, Hungary*
[2, 3, 4]*Institute Computer Science and Control, Hungarian Academy of Sciences, Budapest, Hungary*

Abstract. A full methodology of short-term traffic prediction is proposed for urban road traffic network via Artificial Neural Network (ANN). The goal of the forecasting is to provide speed estimation forward by 5, 15 and 30 min. Unlike similar research results in this field, the investigated method aims to predict traffic speed for signalized urban road links and not for highway or arterial roads. The methodology contains an efficient feature selection algorithm in order to determine the appropriate input parameters required for neural network training. As another contribution of the paper, a built-in incomplete data handling is provided as input data (originating from traffic sensors or Floating Car Data (FCD)) might be absent or biased in practice. Therefore, input data handling can assure a robust operation of speed forecasting also in case of missing data. The proposed algorithm is trained, tested and analysed in a test network built-up in a microscopic traffic simulator by using daily course of real-world traffic.

Keywords: urban traffic, pattern recognition, short-term forecasting, average speed, artificial neural network.

Introduction

The forecasting of traffic states has always been a popular topic in transportation research. Several investigations have been conducted both for freeway and urban traffic parameter prediction, such as traffic flow, travel time, occupancy, probability of congestion, or emission (Vlahogianni *et al.* 2014; Zefreh, Török 2016; Buzási, Csete 2015). In our days, especially short-term road traffic prediction has become an important problem due to the new mobility trends, i.e. emerging ITS (Intelligent Transportation System) tools for traffic management as well as sharing economy in transport. The importance of relevant average speed forecast is straightforward for ITS applications, e.g. (Csikós *et al.* 2015a; Ficzere *et al.* 2014). At the same time, it must be emphasised that resources' sharing based services also require information sharing. In fact, resource pooling in urban transport cannot be successfully achieved without appropriate knowledge of real-time and future traffic states.

Traffic prediction methods can be classified as classical prediction or data-driven methods. Classical methods apply micro- and macroscopic traffic models and/or use statistical tools for model-based estimation, e.g. Ben-Akiva (1998), Van Grol *et al.* (1999) or Lin *et al.*

(2008). Concerning the classical methods with statistical approach, several methods can be referenced: Bayesian network models (Fei *et al.* 2011), History Average (HA) models, Autoregressive Integrated Moving Average (ARIMA) models (Williams *et al.* 1988; Billings, Yang 2006; Guin 2006), non-parametric regressions are most commonly used for prediction, as well as procedures based on Kalman filter (Okutani, Stephanedes 1984; Guo *et al.* 2014). These prediction methods achieve their forecast through the analysis of historical data time series. Therefore, they are mostly used for freeway traffic. For this reason, with the evolution of computational intelligence, data-driven methods have gained attention, which consider the traffic system as a black box. This approach is based on analysing the data in order to find relations between the input and output state variables. Data-driven methods may also be complemented by statistical tools to improve future predictions. One of the most popular data-driven techniques is the self-learning pattern recognition based on Artificial Neural Network (ANN) models (Vlahogianni *et al.* 2005; Dougherty, Cobbett 1997; Chen *et al.* 2012), fuzzy-rule based logics (Li *et al.* 2008), Support Vector Machines (SVM) (Yu

---

*Corresponding author. E-mail: *tettamanti@mail.bme.hu*

Taylor & Francis
Taylor & Francis Group

et al. 2013), k-means clustering (Montazeri-Gh; Fotouhi 2011; Lin *et al.* 2013), and expectation maximization based algorithm (Lo 2013). Summarizing the contributions of the papers cited above, the main advantage of the data-driven methods is the ability to capture the linkage of traffic variables in complex urban network, even under rapidly changing conditions.

A recent overview of the latest traffic forecasting research and future challenges are provided by Vlahogianni *et al.* (2014). Therefore, a fully detailed literature review is omitted now. Only the most relevant papers are discussed by focusing on the new contributions proposed in this paper.

In our days, traffic flow estimation via computational intelligence has been deeply investigated. Gastaldi *et al.* (2014) presented a combined ANN-Fuzzy method for estimating the average daily traffic flow based on one-week traffic counts. Zhu *et al.* (2014) also aimed to forecast traffic volume by using radial basis function neural network. Lozano *et al.* (2009) introduced a camera-based image processing method for congestion level recognition without prediction. Srinivasan *et al.* (2009) and Dimitriou *et al.* (2008) proposed an ANN based urban traffic flow forecast. Kumar *et al.* (2013, 2015) investigated short-term traffic flow prediction via ANN and successfully validated the method by using highway traffic flow data. Albeit the previous results clearly justify the applicability of computational intelligence, they only investigate traffic flow prediction. Concerning the travel time or speed forecasting via ANN, several studies are also available. These papers, however, only concern prediction on freeway or urban arterial road, see the papers listed by the review article of Vlahogianni *et al.* (2014), or the papers of Liu *et al.* (2006), Van Lint (2004) or Basu, Maitra (2006). At the same time, the traffic speed forecast for complex urban network with traffic lights is still a rarely investigated problem. After a thorough literature review, only the articles of Fusco *et al.* (2015) and Csikós *et al.* (2015b) have been found as research results to this specific problem. The former one studied short-term traffic prediction problem on large urban network by using Floating Car Data (FCD) via ANN. The latter one also proposed ANN based prediction for speed categories in an arbitrary urban network.

As compared to the surveyed relevant papers, the contributions of our research are listed below providing solutions to some open problems:

- data-driven methods might be very sensitive to training data quality. This fact is especially important as input parameters in this problem can be generated form FDC or traffic sensors. In practice, however, this kind of measurement data is often biased or intermittent by nature. Therefore, the ANN might suffer a loss on performance in case of training based on incomplete data. The previously cited research works do not deal with the missing data problem, or apply a preprocessing step known as imputation to generate missing values (e.g. by using mean). The application of these techniques, however, may reduce the final prediction performance in case of absent data.

Therefore, a built-in incomplete data handling is proposed as in Viharos *et al.* (2002) in order guarantee a robust operation of speed forecasting also in case of missing data;

- the choice of the right input parameters for neural network training is not a straightforward problem. The reviewed papers generally apply a few input parameters without any preprocessing. As a conscious approach, this paper suggests an additional algorithm called feature selection in order to find the proper set of inputs (statistical parameters). The best operation for machine learning is typically achieved by using small set of input variables, the feature selection is crucial because it allows defining a wide range of features which can be later filtered based on their importance thus reducing the number of input variables for model building;

- due to the lack of appropriate (frequent) real-world speed data, data originating from validated microscopic traffic simulator was used during the research. This, however, resulted in an additional contribution. The simulation based data generation is capable producing a huge set of traffic states, moreover extreme scenarios, which only occur rarely. Consequently, compared to a real-word traffic measurement it has the great advantage, i.e. the forecasting model is trained to cover a much wider range of traffic situations;

- a detailed comparative analysis has been carried out on the performance of two methods, the suggested ANN approach and the widely used SVM. These machine learning methods are very popular due to their generalization capabilities and modelling ability of complex nonlinear problems with high accuracy. They are especially useful for modelling of highly uncertain problems where the connections between the variables are unknown;

- finally, it is emphasised that present paper provides a full methodology by summarizing practical considerations on the use of ANN with built-in missing data handling, and thus shares experiences with the research community on the special problem of short-term traffic speed estimation for signalized urban road links.

Basically, the contributed methodology of the paper is motivated by the complex engineering problem of short-term traffic state estimation in case of incomplete traffic data. This consists of the following practical tasks: data collection (adequate traffic simulation or measurement), preparation of data, model building (feature selection, ANN training).

Section 1 introduces the basic preparatory tasks for pattern recognition. In Section 2, relevant soft computing techniques and algorithms are presented. Section 3 provides a full methodology for ANN based short-term speed prediction, demonstrated by a traffic simulation based case study. Section 4 contains study to demonstrate the viability of the proposed method. Conclusions are given at the end of the paper.

1. Preparatory works for pattern recognition

The efficiency of pattern recognition can be considerably enhanced by preparing and using appropriate data. During the construction of the dataset, three main aspects are addressed:

– realistic patterns are needed, but recurring patterns must be excluded to avoid overfitting of the ANN;
– the irrelevant data need to be filtered;
– dynamic characteristics of the process need to be built into database.

The first point is realized by creating traffic excitations as a sum of sinusoids with different frequencies. By using this scheme, occurrence of different traffic demand waves can be mimicked (e.g. the short rush before school opening during the morning rush-hour). The amplitudes of the different sinusoids are given by random variables to exclude deterministic patterns.

The second point is addressed considering topologic characteristics: spatially irrelevant information (i.e. the data of non-connected links) is excluded. When creating the database, it is reasonable to exploit the dynamic characteristics of the system. The most basic consideration is that the analysis and prediction horizon need to be longer than the time constant of the system. Further dynamic characteristics can be involved by using statistical features, such as high-order moments, the tendencies and the highest relative variations. The additional input features of the neural network are tabulated into Table 1. Note that the prediction is carried out for each link separately, thus for each link a dedicated dataset must be calculated.

Table 1. The applied statistical features

Notation	Description	Formula
1st moment	mean	$\bar{x} = \frac{1}{n} \cdot \sum_{k=1}^{n} x_k$
2nd moment	variance	$\frac{1}{n} \sum_{k=1}^{n} (x_k - \bar{x})^2$
3rd moment	skewness (asymmetry in data distribution)	$\frac{1}{n} \sum_{k=1}^{n} (x_k - \bar{x})^3$
4th moment	kurtosis (peakedness in data distribution)	$\frac{1}{n} \sum_{k=1}^{n} (x_k - \bar{x})^4$
Max	maximum value	$\max_{k} x_k$
Min	minimum value	$\min_{k} x_k$
Max/Min	ration of maximum and minimum	$\dfrac{\max_{k} x_k}{\min_{k} x_k}$
Max–Min	difference of maximum and minimum	$\max_{k} x_k - \min_{k} x_k$
Tendency	tendency	$\mathrm{sign}\left(x_n - x_1\right)$

2. ANN model for pattern recognition

This section describes those soft computing techniques and algorithms that are applied in the presented methodology.

2.1. First stage: feature selection

The Euclidian-distance based feature selection algorithm was originally proposed by Devijver and Kittler (1982) and it assumes a pure classification task with the goal of reducing the number of inputs needed for one single output. As a generalization, continuous output parameters can be mapped onto the discrete classification scheme with an appropriate heuristics. Such heuristics are used in the applied method, where the values of the output encountered in the training data set are grouped into the highest possible number of clusters (i.e. intervals of equal length), so that at least one element is contained in each interval.

Once the continuous output vector is transformed into a discrete range, the feature selection algorithm can be applied to rank the input features based on relevance. This is done by using sequential forward selection and applying a statistical measure, which tries to maximize the separability of the output classes. The following equations define the statistical measure:

$$M = \frac{S_b}{S_w}; \qquad (1)$$

$$S_b = \sum_{i=1}^{c} \frac{n_i}{n} \cdot \left(\underline{m}_i - \underline{m}\right) \cdot \left(\underline{m}_i - \underline{m}\right)^T; \qquad (2)$$

$$S_w = \sum_{i=1}^{c} \frac{n_i}{n} \cdot \frac{\sum_{j=1}^{n_i} \left(\underline{p}_{ij} - \underline{m}_i\right) \cdot \left(\underline{p}_{ij} - \underline{m}_i\right)^T}{n_i}, \qquad (3)$$

where: c is the number of classes of the output; n_i is the number of samples in the i-th class; n is the number of samples; $\underline{m}_i$ is the centre of gravity of the i-th class; $\underline{m}$ is the centre of gravity of the samples; $\underline{p}_{ij}$ is the j-th sample of the i-th class.

Vector parameters $\underline{p}_{ij}$, $\underline{m}_i$ and $\underline{m}$ are defined in a subset of the whole feature set, i.e. the dimension of these vectors equal to the number of features the subset contains. The dimension of $\underline{p}_{ij}$, $\underline{m}_i$ and $\underline{m}$ is increasing over the iterations of the sequential forward selection as more and more features are selected. In a given iteration the newly selected feature is the one where the M value of the containing subset is the highest. Basically S_b represents the average distance between the classes and S_w represents the average distance within the classes and M has to be maximized in each iteration for the classes to be the most separated in a given subset.

Figure 1 describes the pseudocode for the feature selection algorithm, where the *calculate*() function calculates the M value described in Equation (1) on its input set. The set R grows with one new feature in each iteration based on the M value. Note that the number of iteration in which the feature was chosen is also stored in R by pairing it with the feature. This also means that the number of maximization tasks in this algorithm is equals to the number of total features.

```
F <- all features                              do
R <- {}
o <- 1                                         {
DO
        b <- {}                                    for all the not selected features
        v <- 0                                     {
        FOR every f in F
                v_i <- calculateM(R ∪ f)               calculate M for the union of this feature and the
                IF v_i > v                             already selected features
                        b <- f                     }
                        v <- v_i                   let the selected features be the feature set where M is
                END                                maximal
        END                                        }
        R <- R ∪ (b,o)                         while all features are selected
        F <- F / b
        o <- o + 1
WHILE F is not empty
```
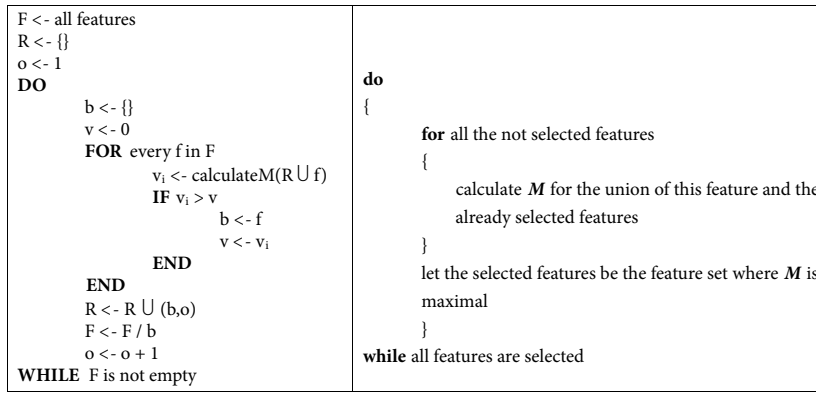
Figure 1. Pure (left) and textual (right) pseudocode of the feature selection algorithm

2.2. Second stage: ANN model building

Over the decades, ANNs proved to be powerful computational models for solving complex estimation and classification problems. An ANN implements the functionality of the biological neural networks (McCulloch, Pitts 1943). One of the most popular and widespread ANN models is the Multi-Layer Perceptron (MLP) (Werbos 1974).

Figure 2 shows an MLP model where the neurons are organized into layers and each layer is fully connected with the next one. Supervised training of an MLP means repeated adjustment of the weight of each link to receive more and more favourable output on specific neurons (output neurons) while stimulating other neurons (input neurons). T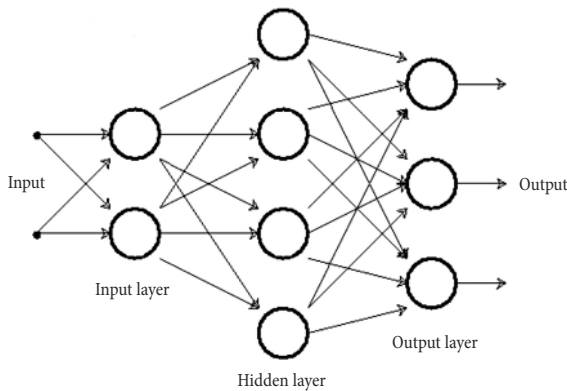he backpropagation algorithm achieves this by calculating the derivatives of the network's error with respect to all of its weights and adjusting the weights to a position where, based on the derivatives, the error is smaller, e.g. moving the weights in the direction of the descent of the derivatives where the error is a measure of the difference between the network's output and the target values for the same input. This is a form of supervised learning because the data samples used for training are known before the model building.

2.3. Built-in incomplete data handling

Incomplete data is a common problem in pattern recognition. The typical solution is to impute the missing or incorrect values with a default or interpolated value. This solution has the downside of generating distortion in the dataset. The applied MLP model has an extension to the original backpropagation algorithm, which allows dynamic handling of missing values (e.g. FCD may be intermittent) (Viharos *et al.* 2002). The concept of the extension is reconfiguring the network for each sample and turning off the input and output neurons corresponding to the missing values. The neurons that are turned off and every link connected to them behave as objects outside the network.

Figure 3 shows the pseudocode for the built-in incomplete data handling, where *w* is the weight vector of the model, *o* is the model output vector and d is the vector of the derivatives respect to the weights. The *forward*() function calculates o for a given *s*, e.g. applies the model on *s*. The *backward*() function computes *d*. The *calcu-*

Figure 2. The MLP model

```
S <- all samples                              do
w <- ann weights                              {
Δw <- {0,...,0}                                   for all the input-output vector pairs
DO                                                {
        FOR every s in S                              turn off the network neurons according to the
                reconfigureann(s)                     missing values of the given data vector,
                o <- forward(w,s)                     forward (based on the input data vector),
                d <- backward(w,o,s)                  calculate the derivatives of the network weights,
                Δw <- Δw + calculatedeltaweights(d)   calculate the corresponding changes of the weights
                resetann()                            and sum them up,
        END                                           turn on all the neurons that were turned off
        w <- w + Δw                               }
WHILE terminated() is false                       change the weights with their corresponding sum
                                                  }
                                              while a special criterion, e.g. the value of estimation error
                                              is higher than required
```
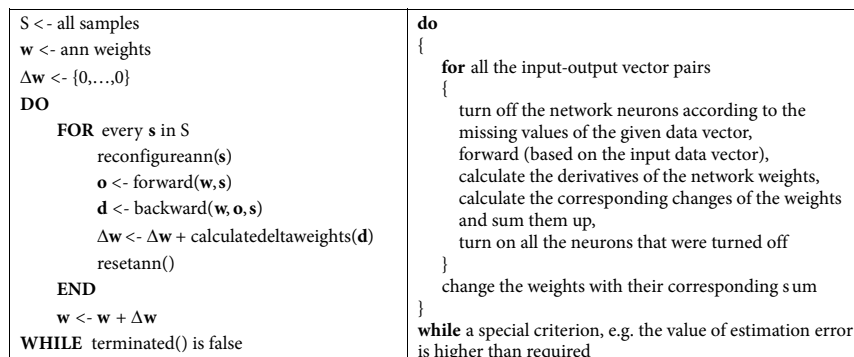
Figure 3. Pure (left) and textual (right) pseudocode of the incomplete data handling in the training algorithm

latedeltaweights() function calculates the weight change Δ*w* based on *d*. The *reconfigureann*() function saves the current configuration of the network and reconfigures it according to the incomplete values in *s*, e.g. excludes the input/output neurons from the network where there are missing values in *s*. While the *resetann*() function restores the original network configuration as if there were no missing values. Finally the *terminated*() function tests if one of the termination criteria is reached, i.e. the maximum number of iteration is reached or the error is below the predefined threshold. Basically, it works the same way as the original backpropagation algorithm, but before the forward and backward calculation the network is reconfigured according to the missing values and after the procession the network is reverted to its original state. In each training iteration for every training sample the following steps are executed:

1) turn off the network neurons according to the missing values of the given data vector;
2) apply the model on the complete part of the input data vector;
3) calculate the derivatives of the network weights;
4) calculate the corresponding changes of the weights and sum them up;
5) turn on all the neurons that were turned off in step 1.

Earlier results (Viharos *et al.* 2002) showed this solution performs better than the typical imputation methods because of the fact that no distortion is added to the data during the procedure. This paper also compares some imputation methods with the built-in data handling concluding the same results.

3. Methodology through a traffic simulation based case study

In the case study, the objective is to predict the state of traffic around a high capacity intersection. The measured data covers only the mean speed of traffic for the network links with a sampling period of 5 min. Based on a measurement record of 30 min long periods, state prediction is carried out for different horizon lengths (5, 15 and 30 min). Using continuous input values, continuous speeds of traffic are forecast.

3.1. Traffic simulation in the test network

The case study network models the vicinity of Oktogon square in District 6, Budapest (Figure 4).

The models were trained and tested using VISSIM simulation data exclusively as we only have limited access to real-word traffic data. The characteristics of the real-world traffic flow (peak period dynamics) were imitated in the simulations. Therefore, daily pattern of traffic demands could be reproduced. On the other hand, the network traffic parameters (e.g. signal plans) are also tuned according to the real-world attributes.

All links are examined in all directions, respectively. The selected road links are separated by intersections with traffic lights. Thus, the length of links are different: the shortest is approximately 100 m (No 10 and 15 in Figure 4), while the longest is approx. 330 m (No 11 and 14).

For the simulations, the microscopic traffic simulator VISSIM is utilized (Wiedemann 1974) together with to MATLAB (Tettamanti, Varga 2012).

3.2. Preparatory works

During a simulation, the mean speed of traffic in each link is measured with a sampling time of 5 min. A sample is given with a row vector, containing the mean speeds of the links. The measurements are organized in 60 min blocks. Thus, one record contains data of 12 measurements. Each record is divided to two parts: the measurement data of the first 30 min are used as inputs (which is further modified), while the last 30 min serve as the basis of outputs of the neural network. Applying a time-shifted framework for the measurement dataset, from a *t*-hour long simulation a total of $(t-1) \cdot 12$ re-
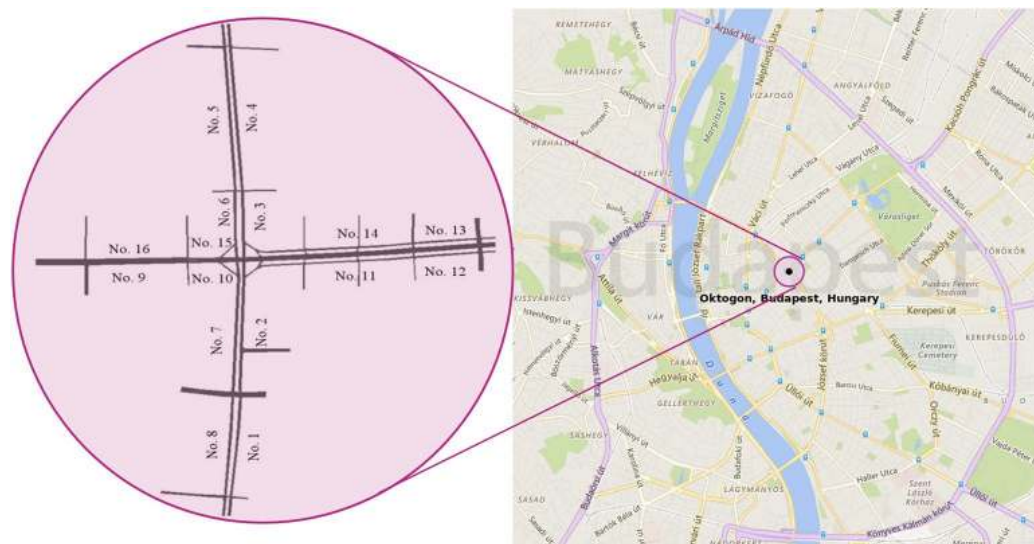

Figure 4. Scheme of the modelled real-world road network (Oktogon square, Budapest, Hungary, GPS: 47.505207, 19.0633920) and bing map of Budapest

cords can be produced. In the case study, 60 simulation runs were conducted, each of them lasting 6-hour long. Each simulation run resulted in 60 records. Therefore, a total of 3600 records were obtained, of which 2500 records were used for training and 1100 for testing the neural network.

In the methodology, the prediction is carried out separately for each link, using dedicated dataset (note that Section 4 presents a case study for link 13 only). First, the non-relevant data are excluded: link measurements of opposite directions (e.g. for link No 1, data of links No 5–8 are excluded). Then, the statistical features of Table 1 for each relevant link are calculated and attached to the input vector. As a result of the preparations, one record of the data set is a vector of length 120.

The output of each pattern recognition problem is thus a three-element continuous valued vector (with state prediction of 5, 15 and 30 min ahead of the last input).

3.3. Application of the ANN

The applied model building method consists of two process stages. The first stage is a feature selection method, which greatly reduces the number of parameters making it applicable for neural network training. In the presented case, there are 120 possible input features describing the whole traffic network and 3 output features as one of the 16 links are estimated over 5, 15 and 30 min in the future. For each output a separate feature selection is required as the order of features depends on the estimation task. Once the order of features is established one can select the best n features for being the inputs of the MLP model. This decision is usually based on expert knowledge. In the presented application the first 10 features are selected in each of the estimation tasks for the sake of comparability. Moreover, the rest of the features can be considered insignificant based on the feature selection measure.

Table 2 shows the first 10 selected features. In the naming, the first part describes the feature type discussed earlier with one exception: Speed means the link mean speed measurements. The second part refers to one of the 16 links where the feature was calculated from. The third part denotes which 5 min (e.g. fifth) of the 30 min input interval is used by the feature (if the third part is missing then the features is calculated based on the whole 30 min interval).

For evaluation purposes, two other methods were tested for selecting features: the mRMR (Maximum-Relevance Minimum-Redundancy) (Peng *et al.* 2005) and the expert knowledge. The expert knowledge means the expert opinion of the engineers or scientists who have significant experience in the field of traffic network dynamics. The selected features are listed in Tables 3–4 and the performance evaluation is presented in Section 4.

Table 2. The selected input features of link 13 using the Euclidian-distance based feature selection

	5 min estimation	15 min estimation	30 min estimation
1	Speed_segment_no_14_sixth_5_min	Speed_segment_no_14_sixth_5_min	1st_moment_segment_no_16
2	Speed_segment_no_13_sixth_5_min	Speed_segment_no_13_sixth_5_min	Min_segment_no_16
3	3rd_moment_segment_no_15	3rd_moment_segment_no_15	1st_moment_segment_no_15
4	3rd_moment_segment_no_16	3rd_moment_segment_no_16	Speed_segment_no_16_sixth_5_min
5	3rd_moment_segment_no_2	3rd_moment_segment_no_2	3rd_moment_segment_no_15
6	3rd_moment_segment_no_1	3rd_moment_segment_no_1	Speed_segment_no_16_fifth_5_min
7	4th_moment_segment_no_16	4th_moment_segment_no_16	Max_segment_no_16
8	Speed_segment_no_14_fifth_5_min	Max_segment_no_16	Speed_segment_no_16_fourth_5_min
9	3rd_moment_segment_no_13	Speed_segment_no_16_sixth_5_min	Max_segment_no_15
10	Speed_segment_no_13_fifth_5_min	Speed_segment_no_15_sixth_5_min	Speed_segment_no_16_third_5_min

Table 3. The selected input features of link 13 using mRMR feature selection

	5 min estimation	15 min estimation	30 min estimation
1	Speed_segment_no_15_fifth_5_min	1st_moment_segment_no_3	Speed_segment_no_4_fifth_5_min
2	Speed_segment_no_16_fifth_5_min	Speed_segment_no_4_fifth_5_min	Speed_segment_no_4_sixth_5_min
3	Speed_segment_no_14_sixth_5_min	Speed_segment_no_4_sixth_5_min	3rd_moment_segment_no_4
4	1st_moment_segment_no_14	3rd_moment_segment_no_4	Speed_segment_no_2_fifth_5_min
5	2nd_moment_segment_no_14	Speed_segment_no_3_fifth_5_min	2nd_moment_segment_no_3
6	4th_moment_segment_no_14	2nd_moment_segment_no_3	1st_moment_segment_no_4
7	Tendency_segment_no_14	4th_moment_segment_no_3	4th_moment_segment_no_4
8	3rd_moment_segment_no_14	1st_moment_segment_no_4	Speed_segment_no_3_fifth_5_min
9	Tendency_segment_no_15	2nd_moment_segment_no_4	4th_moment_segrent_no_3
10	Speed_segment_no_3_fourth_5_min	4th_moment_segment_no_4	2nd_moment_segment_no_4

Table 4. The selected input features of link 13 based on expert knowledge

	5 min estimation	15 min estimation	30 min estimation
1	1st_moment_segment_no_14	1st_moment_segment_no_14	1st_moment_segment_no_14
2	2nd_moment_segment_no_14	2nd_moment_segment_no_14	2nd_moment_segment_no_14
3	1st_moment_segment_no_15	1st_moment_segment_no_15	1st_moment_segment_no_15
4	2nd_moment_segment_no_14	2nd_moment_segment_no_14	2nd_moment_segment_no_14
5	1st_moment_segment_no_3	1st_moment_segment_no_3	1st_moment_segment_no_3
6	2nd_moment_segment_no_3	2nd_moment_segment_no_3	2nd_moment_segment_no_3
7	1st_moment_segment_no_13	1st_moment_segment_no_13	1st_moment_segment_no_13
8	2nd_moment_segment_no_13	2nd_moment_segment_no_13	2nd_moment_segment_no_13
9	1st_moment_segment_no_14	1st_moment_segment_no_14	1st_moment_segment_no_14
10	2nd_moment_segment_no_14	2nd_moment_segment_no_14	2nd_moment_segment_no_14

Table 3 shows the feature selection order using the mRMR algorihm while Table 4 shows 8 features which were selected by using expert knowledge (these were applied for all three tasks). After selecting the most significant features from the feature set during the first stage, the second stage can apply the MLP training on the reduced dataset. For each estimation task a separate model is built and tested, where the inputs of the given model are the 10 selected features.

Furthermore, different versions of each dataset are created for simulating varying amount of incompleteness of the data. Two types of incomplete data have been generated. In the first case, link measurements of random samples are missing. In the second case, the measurements of the whole network are missing from random measurement periods. Incomplete databases are created offline, following a random choice on sample loss. The additional statistical features are then recalculated considering the incomplete measurement data. The following levels of incompleteness are considered: 10, 20 and 50%. This incompleteness is handled by the MLP model automatically and for the SVM comparison imputation is applied on the datasets.

4. Evaluation

The following case study describes the evaluation results for link 13 solely (see the network in Figure 4). Presentation of the estimation results are divided into two parts. The first part shows a comparison of the ANN and SVM models on complete datasets. Then, the second part discusses the performance of these two model types on incomplete datasets. As a state-of-the-art technology, the SVM is chosen for comparative evaluation. SVM is a popular method of our days, widely used in many classification and regression problems (Byun, Lee 2002; Moguerza, Muñoz 2006). One of the most popular SVM library is LIBSVM, which is coded in C++ programming language and has an interface for MATLAB.

The main tuning parameters for the SVM were applied as follows:
- SVM type: nu-SVR;
- Kernel type: radial basis function;

- epsilon: 0.001;
- cost: 1;
- gamma: 0.01.

They were chosen by experimenting with different settings and choosing the one that yields the best performance results.

4.1. Feature selection comparison

This section presents the results of the comparison of three different feature selection approaches: the Euclidian-distance based feature selection described in Section 2.1; the mRMR algorithm (Peng *et al.* 2005) and manual selection using expert knowledge (as described in Section 3.3).

Table 5 shows how the three different feature selection approaches performed after training the MLP model with the selected features. In the 5 and 15 min estimation tasks the models trained with the inputs selected by the Euclidian-distance based feature selection performed better than in the case of the other two approaches. In the 30 min estimation task expert knowledge was the best and the mRMR is the worst but the performance differences are smaller than in the case of the other two tasks. The following analysis (in Section 4.2 and 4.3) was carried using the inputs provided by the Euclidian-distance based feature selection (Table 2).

Table 5. Feature selection comparison considering the traffic speed forecast estimation error [%]

		Feature selection method		
		Euclidian-distance	mRMR	Expert knowledge
Estimation horizon	5 min	11.43	12.11	16.54
	15 min	20.39	27.78	22.22
	30 min	27.95	30.62	25.68

4.2. Speed estimation: full data case

This subsection presents the results of the 3 estimation tasks (5, 15 and 30 min) using the ANN and the SVM model. Both models were trained with the same dataset,

but evaluated on two different test dataset (denoted as *Test data #1* and *Test data #2*) in order to provide a valid comparison. Table 6 shows an overall comparison of the ANN and SVM models. The results show that the ANN provides an estimation of 10–15% lower relative errors compared to SVM.

Figure 5 provides the estimation results of the ANN and SVM models on the 5-min estimation task. The figures display the estimated value for every sample of the test dataset ordered by the real value. The results highlight that the ANN approach performed better than the SVM method. ANN provides the lowest modelling error at low speeds. Also, at high speeds, low relative errors are present. However, during transient periods (in the interval of [15, 40] km/h) high relative errors can be observed. SVM results in a highly uncertain estimation with strong errors generally.

Table 6. Comparison of the MLP and SVM models on two different traffic simulation dataset considering the traffic speed forecast estimation error [%]

		ANN model		SVM model	
		Test data			
		#1	#2	#1	#2
Estimation horizon	5 min	12.11	10.70	26.22	25.25
	15 min	21.18	19.49	31.37	29.89
	30 min	28.89	27.09	42.35	42.46

4.3. Speed estimation: incomplete data case

This subsection discusses the estimation results of the test cases with different amount of incompleteness. As it was mentioned earlier the ANN model has built-in incomplete data handling Viharos *et al.* (2002). The SVM model, however, is unable to perform on an incomplete dataset. For this reason the SVM model was trained and tested on preprocessed data where the missing values were imputed. Three different imputation values are used:

– 0, which is a standard imputation value;
– 0.5, which is the centre of the normalization interval;
– the average value, which is the average value of the not missing values of a given feature.

There are two type of incomplete data generation (both for the training and testing datasets) which were described at the end of Section 3.3. These are referenced as *Incompleteness #1* and *Incompleteness #2* in the Table 7 and Figures 6–8. The selected incompleteness percentages are chosen to represent real-world-like data loss situations, i.e. gradual incompleteness cases of 10, 20 and 30%.

The estimation results based on incomplete data are summarized in Table 7. Note that the imputation value denoted by '–' (in the 3rd column of Table 7) represents that no imputation was applied but instead the ANN with the built-in incomplete data handling.

The ANN approach together with the built-in incomplete data handling results in the best performance in comparison with the other methods. By analysing SVM results, it is observable that with the imputation of 0.5 value and average value the SVM models perform similarly, while the imputation of 0 value shows slightly worse accuracy. The reason for this is that 0 is completely independent from the generated dataset, and while 0.5 is the centre of the normalization interval, the average value trivially depends on the dataset. In conclusion, it can be seen that numerical test results clearly justify the efficient applicability of the proposed method.

Figure 6 depicts a comparison of the ANN and SVM models on the first type of incomplete datasets. In the case of the SVM model only the results with the imputation of average value are shown in the figure as these proved to be the best. It can be seen how the accuracy decreases as the ratio of incomplete data rises in the datasets.

Figure 7 shows a further analysis of the built-in incomplete data handling. In this case, the ANN model is trained on the complete dataset, but evaluated on the incomplete test datasets. The results indicate that the performance of the trained model decreases as the incompleteness increases in the test datasets. The explanation is that the model was not prepared to deal with incomplete data, as it is trained on a complete dataset. Compared to the results depicted in Figure 6, one can observe that if the model is trained with the same amount of incompleteness as in the test dataset, the test performance is much better.
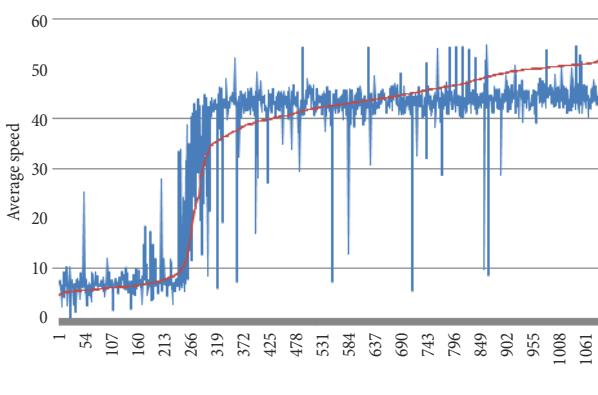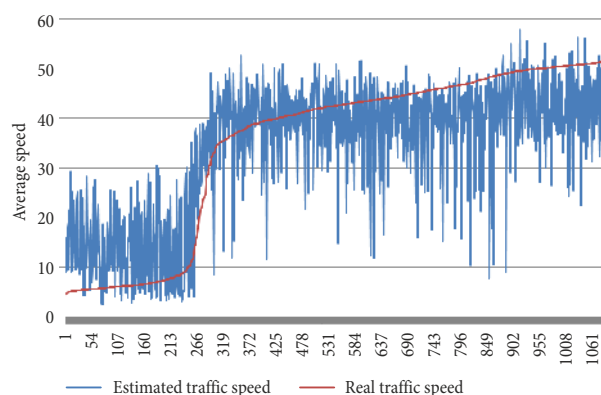

Figure 5. 5-min estimation results on link 13 using the ANN (left) and SVM (right) model
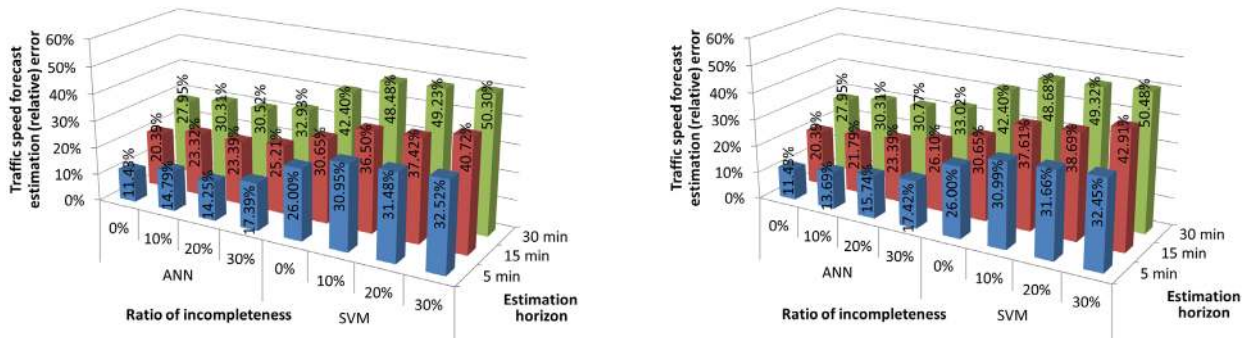
Figure 6. Estimation accuracy of MLP and SVM models on the incomplete datasets
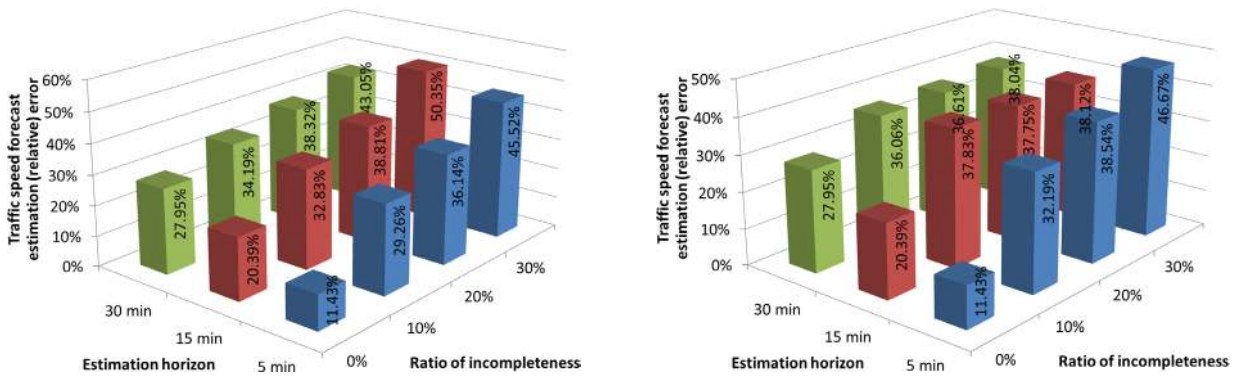(left: *Incompleteness #1*, right: *Incompleteness #2*)

Figure 7. Estimation accuracy of ANN model on different incomplete datasets (trained here on complete dataset)
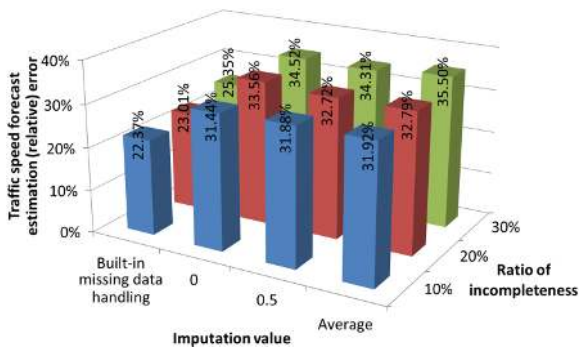(left: *Incompleteness #1*, right: *Incompleteness #2*)

Figure 8. Estimation accuracy of the ANN model with different imputation methods on different incomplete datasets
(data of *Incompleteness #1* and *Incompleteness #2* are considered together)

Figure 8 shows the average estimation capability of the different imputation methods on different level of incompleteness. It can be seen that the built-in missing data handling performs better than the plain imputation methods improving the estimation accuracy with about 10%.

As a summary of the performance analysis and experiences, an engineering suggestion can be given, i.e. the proposed method is fully appropriate for short-term prediction of 5–15 min as simulation results show that the estimation error in this case remains in a reasonable low range. Considering this performance, the elaborated method can be accepted for further use in ITS applications, e.g. traffic incident detection, route guidance, or traffic control.

Conclusions

A traffic speed prediction algorithm and related methodology has been investigated specifically for urban road traffic networks. During the research, important experiences have been gained concerning the methodology for input-output parameter selection and appropriate feature selection.

Numerical results attest that the generation and narrowing of input dataset plays a key role in the urban traffic speed estimation performance.

Basically, two main targets have been achieved. On the one hand, the applicability of the proposed feature selection method was approved. Based on the order of features established by the algorithm a reduced set of 10 parameters could be selected for each of the estimation tasks as the most relevant inputs for ANN training. On the other hand, the advantage of the built-in incomplete data handling solution for traffic speed estimation was shown in the paper. Nevertheless, there is always limitation in such systems.

In the case of the proposed method if data loss overshoots the level of 30%, the performance start decreasing. In conclusion, the simulations justified the viability of the proposed method considering the obtained recognition rates also in comparison of the concurrent SVM method. Hence, acceptable urban traffic speed prediction can be performed for short periods, serving as a practically applicable method for several traffic applications.

Additionally, the application of paper results may also contribute to efficient collaborative transport services capable considering traffic incidents.

Table 7. Test results: relative errors

Model type	Imputation value	Incompleteness percentage	Incompleteness #1 [%]			Incompleteness #2 [%]		
			5 min	15 min	30 min	5 min	15 min	30 min
ANN	–	0	11.43	20.39	27.95	11.43	20.39	27.95
		10	14.79	23.32	30.31	13.69	21.79	30.31
		20	14.25	23.39	30.52	15.74	23.39	30.77
		30	17.39	25.21	32.93	17.42	26.10	33.02
	0	0	11.43	20.39	27.95	11.43	20.39	27.95
		10	25.25	28.71	42.73	22.63	26.50	42.83
		20	25.11	31.13	43.06	27.35	30.19	44.50
		30	27.49	32.77	45.52	26.08	31.35	43.92
	0.5	0	11.43	20.39	27.95	11.43	20.39	27.95
		10	24.16	28.81	43.43	23.42	27.46	44.00
		20	24.42	31.22	44.37	21.90	30.57	43.82
		30	27.09	32.95	44.77	24.84	33.16	43.02
	Average	0	11.43	20.39	27.95	11.43	20.39	27.95
		10	25.99	29.37	42.95	22.69	26.62	43.91
		20	24.78	31.55	43.95	24.16	29.12	43.16
		30	28.53	32.33	45.10	28.51	32.95	45.57
SVM	0	0	26.00	30.65	42.40	26.00	30.65	42.40
		10	31.01	38.40	49.35	31.15	39.44	49.33
		20	31.56	38.93	49.74	32.00	40.80	49.53
		30	32.52	40.72	50.30	32.45	42.91	50.48
	0.5	0	26.00	30.65	42.40	26.00	30.65	42.40
		10	30.97	36.63	48.59	30.98	37.46	48.73
		20	31.49	37.46	49.15	31.67	38.57	49.34
		30	32.24	38.24	49.76	32.16	40.01	50.35
	Average	0	26.00	30.65	42.40	26.00	30.65	42.40
		10	30.95	36.50	48.48	30.99	37.61	48.68
		20	31.48	37.42	49.23	31.66	38.69	49.32
		30	32.52	40.72	50.30	32.45	42.91	50.48

References

Basu, D.; Maitra, B. 2006. Modeling stream speed in heterogeneous traffic environment using ANN-lessons learnt, *Transport* 21(4): 269–273.

Ben-Akiva, M. E. 1998. DynaMIT: a simulation-based system for traffic prediction and guidance generation, in *TRISTAN III: Triennal Symposium on Transportation Analysis*, 17 June 1998, San Juan, Porto-Rico, 1–14.

Billings, D.; Yang, J.-S. 2006. Application of the ARIMA models to urban roadway travel time prediction – a case study, *2006 IEEE International Conference on Systems, Man and Cybernetics*, 8–11 October 2006, Taipei, Taiwan, 2529–2534. https://doi.org/10.1109/ICSMC.2006.385244

Buzási, A.; Csete, M. 2015. Sustainability indicators in assessing urban transport systems, *Periodica Polytechnica Transportation Engineering* 43(3): 138–145.
https://doi.org/10.3311/PPtr.7825

Byun, H.; Lee, S.-W. 2002. Applications of support vector machines for pattern recognition: a survey, *Lecture Notes in Computer Science* 2388: 213–236.
https://doi.org/10.1007/3-540-45665-1_17

Chen, Y.; Yang, B.; Meng, Q. 2012. Small-time scale network traffic prediction based on flexible neural tree, *Applied Soft Computing* 12(1): 274–279.
https://doi.org/10.1016/j.asoc.2011.08.045

Csikós, A.; Tettamanti, T.; Varga, I. 2015a. Macroscopic modeling and control of emission in urban road traffic networks, *Transport* 30(2): 152–161.
https://doi.org/10.3846/16484142.2015.1046137

Csikós, A.; Viharos, Z. J.; Kis, B. K.; Tettamanti, T.; Varga, I. 2015b. Traffic speed prediction method for urban networks – an ANN approach, in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, 3–5 June 2015, Budapest, Hungary, 102–108. https://doi.org/10.1109/MTITS.2015.7223243

Devijver, P. A.; Kittler, J. 1982. *Pattern Recognition: a Statistical Approach*. 1st edition. Prentice Hall. 480 p.

Dimitriou, L.; Tsekeris, T.; Stathopoulos, A. 2008. Adaptive hybrid fuzzy rule-based system approach for modeling and predicting urban traffic flow, *Transportation Research Part C: Emerging Technologies* 16(5): 554–573.
https://doi.org/10.1016/j.trc.2007.11.003

Dougherty, M. S.; Cobbett, M. R. 1997. Short-term inter-urban traffic forecasts using neural networks, *International Journal of Forecasting* 13(1): 21–31.
https://doi.org/10.1016/S0169-2070(96)00697-8

Fei, X.; Lu, C.-C.; Liu, K. 2011. A Bayesian dynamic linear model approach for real-time short-term freeway travel

time prediction, *Transportation Research Part C: Emerging Technologies* 19(6): 1306–1318.
https://doi.org/10.1016/j.trc.2010.10.005

Ficzere, P.; Ultmann, Z.; Török, Á. 2014. Time–space analysis of transport system using different mapping methods, *Transport* 29(3): 278–284.
https://doi.org/10.3846/16484142.2014.916747

Fusco, G.; Colombaroni, C.; Comelli, L.; Isaenko, N. 2015. Short-term traffic predictions on large urban traffic networks: applications of network-based machine learning models and dynamic traffic assignment models, in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, 3–5 June 2015, Budapest, Hungary, 93–101.
https://doi.org/10.1109/MTITS.2015.7223242

Gastaldi, M.; Gecchele, G.; Rossi, R. 2014. Estimation of annual average daily traffic from one-week traffic counts. a combined ANN-fuzzy approach, *Transportation Research Part C: Emerging Technologies* 47(1): 86–99.
https://doi.org/10.1016/j.trc.2014.06.002

Guin, A. 2006. Travel time prediction using a seasonal autoregressive integrated moving average time series model, in *2006 IEEE Intelligent Transportation Systems Conference*, 17–20 September 2016, Toronto, Canada, 493–498.
https://doi.org/10.1109/ITSC.2006.1706789

Guo, J.; Huang, W.; Williams, B. M. 2014. Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification, *Transportation Research Part C: Emerging Technologies* 43(1): 50–64.
https://doi.org/10.1016/j.trc.2014.02.006

Kumar, K.; Parida, M.; Katiyar, V. K. 2013. Short term traffic flow prediction for a non urban highway using artificial neural network, *Procedia – Social and Behavioral Sciences* 104: 755–764. https://doi.org/10.1016/j.sbspro.2013.11.170

Kumar, K.; Parida, M.; Katiyar, V. K. 2015. Short term traffic flow prediction in heterogeneous condition using artificial neural network, *Transport* 30(4): 397–405.
https://doi.org/10.3846/16484142.2013.818057

Li, Z.; Sun, D.; Jin, X.; Yu, D.; Zhang, Z. 2008. Pattern-based study on urban transportation system state classification and properties, *Journal of Transportation Systems Engineering and Information Technology* 8(5): 83–87.
https://doi.org/10.1016/S1570-6672(08)60041-0

Lin, L.; Li, Y.; Sadek, A. 2013. A k nearest neighbor based local linear wavelet neural network model for on-line short-term traffic volume prediction, *Procedia – Social and Behavioral Sciences* 96: 2066–2077.
https://doi.org/10.1016/j.sbspro.2013.08.233

Lin, S.; Xi, Y.; Yang, Y. 2008. Short-term traffic flow forecasting using macroscopic urban traffic network model, in *2008 11th International IEEE Conference on Intelligent Transportation Systems*, 12–15 October 2008, Beijing, China, 134–138. https://doi.org/10.1109/ITSC.2008.4732567

Liu, H.; Van Lint, H. W. C.; Van Zuylen, H. J. 2006. Neural-network-based traffic flow model for urban arterial travel time prediction, in *Transportation Research Board 86th Annual Meeting*, 21–25 January 2007, Washington DC, US, 1–17.

Lo, S.-C. 2013. Expectation-maximization based algorithm for pattern recognition in traffic speed distribution, *Mathematical and Computer Modelling* 58(1–2): 449–456.
https://doi.org/10.1016/j.mcm.2012.11.004

Lozano, A.; Manfredi, G.; Nieddu, L. 2009. An algorithm for the recognition of levels of congestion in road traffic problems, *Mathematics and Computers in Simulation* 79(6): 1926–1934. https://doi.org/10.1016/j.matcom.2007.06.008

McCulloch, W. S.; Pitts, W. 1943. A logical calculus of the ideas immanent in nervous activity, *The Bulletin of Mathematical Biophysics* 5(4): 115–133.
https://doi.org/10.1007/BF02478259

Moguerza, J. M.; Muñoz, A. 2006. Support vector machines with applications, *Statistical Science* 21(3): 322–336.
https://doi.org/10.1214/088342306000000493

Montazeri-Gh, M.; Fotouhi, A. 2011. Traffic condition recognition using the *k*-means clustering method, *Scientia Iranica: Transactions B: Mechanical Engineering* 18(4): 930–937.
https://doi.org/10.1016/j.scient.2011.07.004

Okutani, I.; Stephanedes, Y. J. 1984. Dynamic prediction of traffic volume through Kalman filtering theory, *Transportation Research Part B: Methodological* 18(1): 1–11.
https://doi.org/10.1016/0191-2615(84)90002-X

Peng, H.; Long, F.; Ding, C. 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(8): 1226–1238.
https://doi.org/10.1109/TPAMI.2005.159

Srinivasan, D.; Chan, C. W.; Balaji, P. G. 2009. Computational intelligence-based congestion prediction for a dynamic urban street network, *Neurocomputing* 72(10–12): 2710–2716. https://doi.org/10.1016/j.neucom.2009.01.005

Tettamanti, T.; Varga, I. 2012. Development of road traffic control by using integrated VISSIM-MATLAB simulation environment, *Periodica Polytechnica Civil Engineering* 56(1): 43–49. https://doi.org/10.3311/pp.ci.2012-1.05

Van Grol, H. J. M.; Danech-Pajouh, M.; Manfredi, S.; Whittaker, J. 1999. DACCORD: on-line travel time prediction, in *World Transport Research: Selected Proceedings of the 8th World Conference on Transport Research*, 12–17 July 1998, Antwerp, Belgium, 455–467.

Van Lint, J. W. C. 2004. *Reliable Travel Time Prediction for Freeways: Bridging Artificial Neural Networks and Traffic Flow Theory*: PhD Thesis. Delft University of Technology, Netherlands. 325 p.

Viharos, Z. J.; Monostori, L.; Vincze, T. 2002. Training and application of artificial neural networks with incomplete data, *Lecture Notes in Computer Science* 2358: 649–659.
https://doi.org/10.1007/3-540-48035-8_63

Vlahogianni, E. I.; Karlaftis, M. G.; Golias, J. C. 2014. Short-term traffic forecasting: where we are and where we're going, *Transportation Research Part C: Emerging Technologies* 43(1): 3–19. https://doi.org/10.1016/j.trc.2014.01.005

Vlahogianni, E. I.; Karlaftis, M. G.; Golias, J. C. 2005. Optimized and meta-optimized neural networks for short-term traffic flow prediction: a genetic approach, *Transportation Research Part C: Emerging Technologies* 13(3): 211–234.
https://doi.org/10.1016/j.trc.2005.04.007

Werbos, P. J. 1974. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*: PhD Thesis. Harvard University, Cambridge, US. 454 p.

Wiedemann, R. 1974. *Simulation des Straßenverkehrsflusses*. Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe, Deutschland (in German).

Williams, B.; Durvasula, P.; Brown, D. 1998. Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models, *Transportation Research Record: Journal of the Transportation Research Board* 1644: 132–141. https://doi.org/10.3141/1644-14

Yu, R.; Wang, G.; Zheng, J.; Wang, H. 2013. Urban road traffic condition pattern recognition based on support vector machine, *Journal of Transportation Systems Engineering and Information Technology* 13(1): 130–136. https://doi.org/10.1016/S1570-6672(13)60097-5

Zefreh, M. M.; Török, Á. 2016. Improving traffic flow characteristics by suppressing shared taxis maneuvers, *Periodica Polytechnica Transportation Engineering* 44(2): 69–74. https://doi.org/10.3311/PPtr.8226

Zhu, J. Z.; Cao, J. X.; Zhu, Y. 2014. Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections, *Transportation Research Part C: Emerging Technologies* 47(2): 139–154. https://doi.org/10.1016/j.trc.2014.06.011