Neural Comput & Applic (1999)8:135–142
© 1999 Springer-Verlag London Limited

Neural
Computing
& Applications

Prediction of Road Traffic using a Neural Network Approach

R. Yasdi

GMD FIT, German National Research Center for Information Technology, Human-Computer Interaction Dept., Sankt Augustin, Germany

A key component of the daily operation and planning activities of a traffic control centre is short-term forecasting, i.e. the prediction of daily to the next few days of traffic flow. Such forecasts have a significant impact on the optimal regulation of the road traffic on all kinds of freeways. They are increasingly important in an environment with increasing road traffic problems. The present paper aims at presenting the effectiveness of a neural network system for prediction based on time-series data. We only use one parameter, namely traffic volume for the forecasting. We employ artificial neural networks for traffic forecasting applied on a road section. Recurrent Jordan networks, popular in the modelling of time series, is examined in this study. Simulation results demonstrate that learning with this type of architecture has a good generalisation ability.

Keywords: Learning; Neural networks; Prediction; Time series

1. Introduction

The *Traffic Control Centre* (TCC) is the core of the traffic control system in Germany. On one side they are connected with data collection systems; on the other side, they serve to control variable message sign systems and to prepare messages for broadcasting. The forecasting of road traffic is an important aspect of traffic management, particularly for traffic information systems, in order to give realistic travel time estimates to road users. Increasing traffic demand has necessitated the extension of the TCC with intelligent tools for the monitoring, control and management of traffic.

Predicting the future behaviour based on a time-series of collected historical data is a serious problem in many scientific applications. A time series is a sequence of time ordered data values that are measurements of a physical process. In the frame of traffic control, loop detectors are the main measurement sources. They mainly provide information on the flow of vehicles. Predictions of these parameters are vital components, particularly for traffic information systems. The prediction of forecasting can be formulated as the estimation of the value of time t_m from the knowledge of values of the same variables, or other process variables, at time $t_{m-l},\cdots t_{m-2},t_{m-1}$, where m is a measure of the forecasting horizon, and l is the measure of the time history necessary to adequately represent the phenomena.

There have been numerous methods for learning and predicting time series, ranging from the traditional time series analysis [1] to recent approaches using neural networks [2]. A central issue common to all of them is the determination of a model structure which substantially affects the performance of learning and prediction of the resulting models. The model with the smallest mean prediction errors is selected from a set of models as the best one. In this way, they give a solution to the problem of model selection. However, generally speaking, the mean square error for training data monotonically decreases as the size of the model increases. On the

Correspondence and offprint requests to: R. Yasdi, GMD FIT, German National Research Center for Information Technology, Human-Computer Interaction Department, 53754 Sankt Augustin, Germany. E-mail: ramin.yasdi@gmd.del

other hand, the mean error for the test data (also referred to as the mean prediction error) decreases in the beginning, but increases again as its size increases. The increase in the mean prediction error is attributed to the increase in the variance of parameters to be estimated. This suggests the existence of an optimal model size which maximises its generalisation ability in terms of the mean prediction error. Therefore, the mean prediction error is not powerful enough to find the best model structure among all the candidates. The model must have the ability of better generalisation.

Jordan neural networks [3], popular in the modelling of time, are examined in this study. For this purpose, a neural network based on a Jordan architecture is designed and trained to predict the future values of the traffic time series using its past values.

The objective of this research is to produce a model capable of producing reliable, consistent and accurate traffic volume forecasting. In the subsequent section, the method of collection and representation of data for this task has been described. This is followed by an overview of recurrent neural network models for a time series. Hereafter, the learning and prediction performance of the model is shown. The paper concludes with a discussion of the method presented, related works, and a look ahead to the extension and future direction of this work.

2. Representation of Data

2.1. Collection of Data

Usually, the traffic data (volume, flow, speed, street condition, climate condition, etc.) are measured by inductive loops in the roads and forwarded to the local road section. The local road section collects the different data units, processes the data aggregation per area and sends the results to the TCC. The data is represented by a time series. A time series is a sequence of time ordered data values that are measurements of a physical process. In the frame of road traffic control, loop detectors are the main measurement sources.

In this particular study, the traffic volume (i.e. number of vehicles passed) are collected every five minutes [4,5]. The data is measured on the left line on the German National Road B39. This road bypasses the Volkswagen company, therefore it is heavily used by VW's employees. Due to the enormous amount of data, a reduction of data is necessary. With a five minute time interval the data oscillates too much, and the random part is high. Therefore, it it necessary to aggregate them to ten or fifteen minute intervals. The mean value is built with three values. To obtain a smooth time curve an average of three values can be computed. The main advantages of this representation are the great reduction in the amount of data in the database, and the easy handling in the simulator.

2.2. Classification of Data

The day of measurement as well as the events which occur at the area can strongly influence the traffic. We distinguish between ordinary days such as Mondays to Sundays, and event days on which a football game or concerts take place. We consider six classes of patterns: for ordinary Monday; Tuesday to Thursday; Friday; Saturday; Sunday and Holidays; and one class of special day for expected events. At present, we do not consider unexpected events. This is a topic for future research. Figure 2 characterises a typical Monday, with the morning peak between five and seven o'clock, a less pronounced afternoon and dilated evening peak. At six in the morning and two o'clock in the afternoon there are shift changes at VW.

If we make the assumption that the traffic pattern reoccurs for similar events, we can predict the traffic flow for the day under the same traffic conditions. Each class has a representative pattern, called the reference pattern, which is stored in a knowledge base and chosen for prediction. Days are represented by absolute patterns, events are related to the absolute values.

2.3. Knowledge Base

The knowledge base contains all the information about the traffic and the network for which the predictions are to be made. It consists of three parts:

1. Verbal information about road works, estimations of the capacity of the road, and a library for all predictable events.
2. Rule parts representing the expertise knowledge in the form of *if* . . . *then* . . . rules. In the latter version, fuzzy rules will constitute this knowledge.
3. Database, containing the pattern and measured values. These data values are normalised from 0 to 1 based on the data of the entire period.

To increase the validity of the prediction, the knowledge base has to be regularly maintained. Once a day the measured data will be transferred to provide the current pattern. To be able to

incorporate this into the knowledge base, we have to build the equivalent historic pattern that can be obtained as follows: according to the events which have been occurring every day, the corresponding historic event pattern that is available in the knowledge base will be updated. The historic pattern obtained and the current day have to be compared, and the mean deviation between the two patterns will be calculated.

3. Partially Recurrent Networks

In this section we shall focus on training a recurrent neural network to recognise or reproduce a temporal sequence of slots. In this regard, we first introduce *partially recurrent networks* [6] which has often been used for sequence recognition and reproduction. These networks are basically backpropagation networks with proper feedback links. In other words, the connections in the partially recurrent networks are mainly feedforward, but include a carefully chosen set of feedback connections. The main function of partially recurrent networks is to deal with time explicitly, as opposed to representing temporal information spatially, as in time-delay neural networks. The recurrence in the partially recurrent network allows the network to remember cues from the recent past, but does not appreciably complicate the structure and training of the whole network. In most cases, the weights on the feedback links are fixed, and so the general backpropagation learning rule may be easily used for training. Such networks are also referred to as *sequential networks* [7], and the nodes receiving feedback signals are called *context units*. In these networks, forward propagation is assumed to occur quickly or without reference time, while the feedback signal is locked. Hence, at time t the context units have signals coming from part of the network state at time $(t-1)$. In this way, the context units remember some aspects of the past, which forms a context for processing at time t. Thus, the state of the whole network at a particular time depends upon an aggregate of previous states as well as on the current input. The network can therefore recognise sequences on the basis of its state at the end of the sequence, or even predict (reproduce) the successor of a given token in a temporal sequence. Since a partially recurrent network is basically a multilayer feedforward network, feedback links can come from either the output nodes or the hidden-layer nodes, and the destinations of the feedback links (i.e. the context units) can be either input nodes or hidden-layer nodes. Because of these various possibilities, there are several different models of partially recurrent networks. Two major models are *Jordan's sequential networks* and *simple recurrent networks*. The former includes Jordan networks [3]. The latter include Elman and buffer networks [8,9].

Elman networks have the potential to master long sequences of time series with the limited means of a learning procedure that is *completely local in time*. Here the feedback links are from the hidden layer to the context layer. The input layer can be considered to be divided into two parts: true input units and context units. Context units simply hold a copy of the activations of hidden nodes from the previous time step. They thus provide the system with memory in the form of a trace of processing in the previous time slice. Like Jordan's sequential networks, the conventional backpropagation method can be applied to train a simple recurrent network.

3.1. Jordan's Sequential Network

This model is realised by adding recurrent links from the network's output to a set of context units C_i, which form a context layer, and from context units to themselves. Figure 1 shows the general structure of the Jordan model. In this figure, single arrows represent connections only from the ith node in the source layer to the ith context unit in the destination (context) layer, whereas wide arrows represent feedforward connections which are usually fully connected, as in the general backpropagation network. In such a network, the output associated with each state is fed back to the context units and blended with the input representing the next state on the input nodes. The whole constitutes the new network state for processing at the next time step.

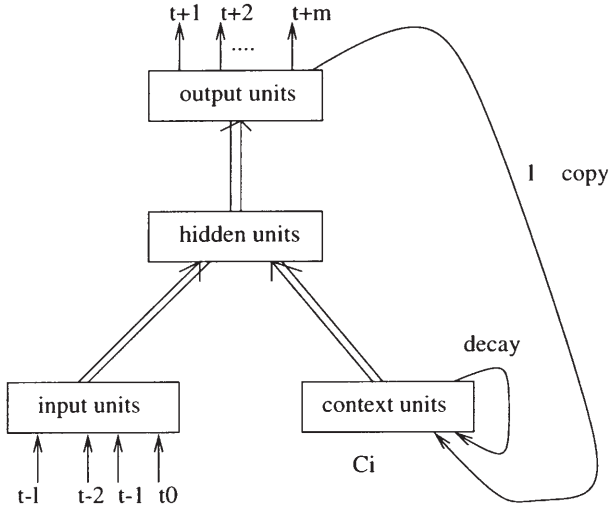

Fig. 1. Structure of the Jordan network.

After several steps of processing, the pattern present on the context units, together with the input units, are characteristic of the particular sequence of the states that the network has traversed. In other words, context units copy the activations of the output nodes from the previous time step through the feedback links with unit weights. The self-connections in the context layer give the context units C_i themselves some individuality or inertia. The model has an input, with the value at time t_0 the most recent value and a set of past values up to the time t_l; the output of the network is composed by the predicted values in the forecasting horizon from t_{+1} to t_{+m}. This allows a richer model with which to represent time series.

It is, in most cases, difficult to obtain *a priori* information on the structure of a network to a given task. This necessitates the learning of various sized networks by trial and error, which is one of the main difficulties in BP learning. BP learning also has difficulty in interpreting hidden units. This difficulty is attributed to the excess degrees of freedom of a network, and to distributed representation on hidden layers. Therefore, it is important to underline that there are no systematic procedures to find the optimal configuration of a neural network (number of hidden layers and neurons). In the first experiment, we tried to find a suitable structure by analysing several network topologies, one network for each day. We then compared the performance of the learning algorithms, then used one single network to predict both traffic volume and accuracy rate. Finally, we came up with a multiple layer Jordan network based on a backpropagation training procedure for this application. The neural network structure consists of two hidden layers of four neurons followed by an output layer of one linear neuron, while the number of input nodes is set to four.

3.2. The Initialisation Functions JE-Weights

The initialisation function for Jordan-Elman-weights requires the specification of five parameters:

- α, β: the weights of the forward connections are randomly chosen from the interval $[\alpha = 1, \beta = -1]$.
- λ: weights of self-recurrent links from context units to themselves. We used $\lambda = 0$.
- γ: weights of other recurrent links to context units. This value is often set to 1.0.
- ψ Initial activation of all context units.

These values are to be set as the initialisation parameters in the simulator.

3.3. Learning Functions

By deleting all recurrent links in a partial recurrent network, a simple feedforward network remains. The context units now have the function of input units, i.e. the total network input consists of two components. The first component is the pattern vector, which was the only input to the partial recurrent network. The second component is a state vector, which is given through the next-state function in every step. In this way, the behaviour of a partial recurrent network can be simulated with a simple feedforward network, that receives the state not implicitly through recurrent links, but as an explicit part of the input vector. In this sense, backpropagation algorithms can easily be modified for the training of partial recurrent networks [10]. The following learning functions are available in the simulator for the partial recurrent network: *Standard Backpropagation*, *Standard Backpropagation with Momentum-Term*, *Quiqprop* and *Rprop*. We have done some comparative studies of these learning functions. However, the variants of backpropagations did not show significant improvement against the standard backpropagation, at least in our example. The parameters for these learning functions are the same as for the regular feedforward versions of these algorithms. The parameters in learning are a learning rate $\eta = 0.02$ and a momentum $\alpha = 0.2$.

The network is designed to predict the traffic volume given the past four values of the time series. A set of 1000 consecutive 15 minute samples was extracted from the data available. This is the volume of 36 days. The set was divided in a training set (4×7 days of week) and a test set (2×7 days of week).

3.4. Model Performance

The model generates a forecast for the next 24 hour period from the daily traffic profile. Figure 2 shows the temporal variation of the target and output traffic volumes for randomly selected days. The network was extended for the whole set of data, and the results were quite satisfactory. In the figure the comparison of the original traffic volume with the neural network predicted values for 24 hours can be seen. As shown, the predicted and measured values are in close agreement.

Evaluation of the model performance can be done by the *Mean Square Error*, $MSE = \Sigma_i |t_i - o_i|^2$, calculated as the difference between forecasted and actual demand. The average errors for the forecasting up to 24 hours are about 0.003. Another parameter of

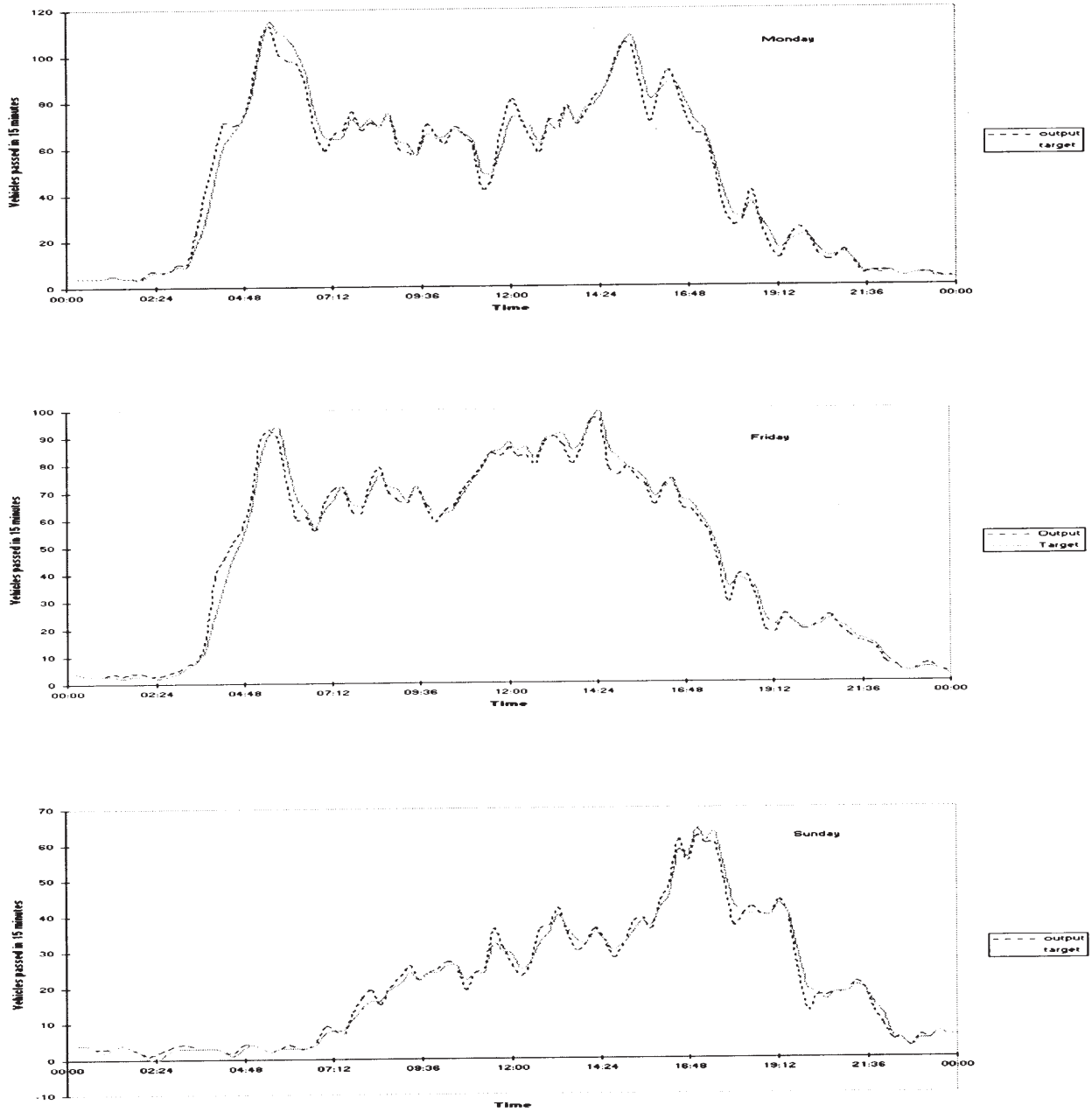

Fig. 2. Temporal variation of the predicted and measured traffic volumes for a randomly selected Monday, Friday and Sunday.

the model performance which could be used is the maximum error in the forecast.

Models, based on Jordan networks demonstrate very good generalisation capabilities. The accuracy of the forecasting is high, even if the network is tested with the data sets that are measured in months later than the data that are used in training. This is particularly interesting because it is an indication that accurate predictive models can be developed without necessarily using adaptive learning.

3.5. Prediction with Different Time Horizons

We distinguish between three forms of predictions, described as follows:

1. *Long-term* prediction. The long-term prediction is a session-oriented procedure that provides prediction for a week. It relies on a historic reference pattern of a week stored in the knowledge base. The sessional information such as summer time

and Christmas vacations will be taken into account here.

2. *Mid-term* prediction. This provides a time horizon of 24 hours. If the day is considered an ordinary day, then it relies on a typical week day stored in the knowledge base. If the day is a predictable event day, then it relies on the predictable event stored in knowledge base. The predicted pattern will then be compared with the current incoming pattern. If the divergence between the historic and current pattern becomes too great, actions have to be taken to control traffic manually.
3. *Short-term* prediction. The short-term predication is based on current incoming data which uses the ability of the neural network for prediction of the future. The first predicted output value is used as one of the lagged inputs for the next prediction into the future. Similarly, the prediction at the second time step, as well the previous time step, are used as lagged inputs for the next prediction three time steps into the future. If the actual prediction value p_i is part of the next input pattern inp_{i+1} to predict the next value p_{i+1}, the pattern inp_{i+1} cannot be generated before the needed prediction value p_i is available. For short-term predictions, many input patterns have to be generated in this manner. To generate these patterns manually means a lot of effort. Using an update function *JE-Special* available on the simulator, these input patterns will be generated dynamically. Let n be the number of input units and m the number of output units of the network. The special function generates the new input vector with the output of the last $n - m$ input units and the output of the m output units. The usage of this function requires $n > m$. Using this function the short-term prediction has a time horizon of about 24 hours. If it is used as online prediction, it starts prediction after a few measured values. This kind of prediction is based on the information on current situations. Figure 4 shows the variation between predicted and present values up to about 24 hours. As the figure shows, after 24 hours there is no change in the output value of the network which means the saturation of the network resulting in no further predication.

3.6. Typical Day Profile

Forecasting is done on a daily basis. An expert forecasts the most critical points during the day. The forecaster selects what he estimates to be the most suitable 'standard day', and then the standard day profile is adjusted with the expert's experience and intuition so as to achieve the profile forecast for the day. The forecaster selects a profile that he considers to be a close approximation to that which is expected for the future period. The basis upon which a profile is chosen as the standard day is made by comparison of the characteristics for the day in question. It is worth noting at this stage that the standard day and the day to be forecasted will, in virtually every case, have the same calendar 'day name'. In the model, the standard day is chosen as the same day the previous week.

Predictable events. When a parameter has a larger or smaller effect than normal, for example a match day, then these events have to be matched as a 'special day'.

3.7. Software Tools used for Control and Management

The analysis shown above requires software to carry out data pre-processing, design of neural networks and a possibility to integrate the solution into the existing traffic control centre. The software packages used and developed are listed as follows:

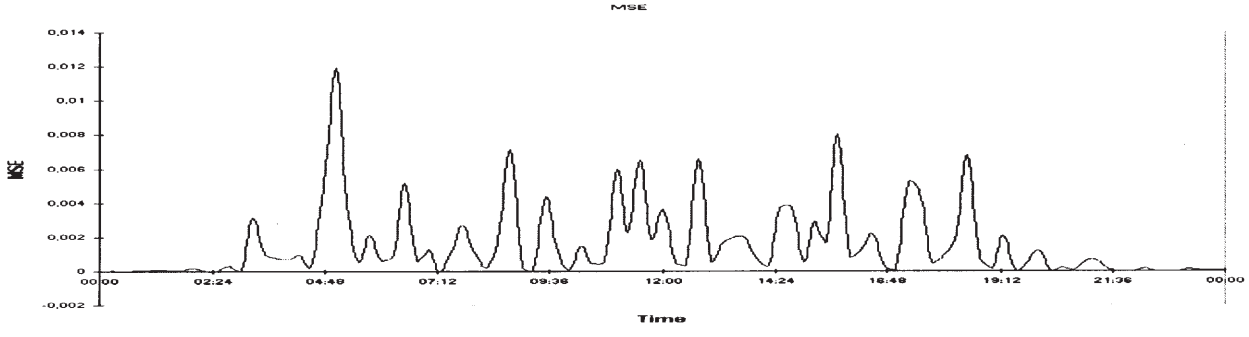

Fig. 3. MSE values of output for a randomly selected day.

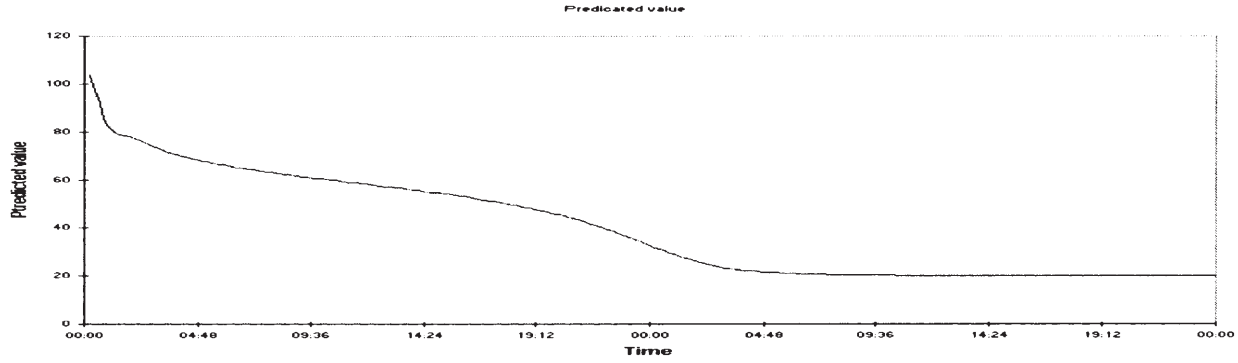

Fig. 4. Variation of used and predicted values in short-term prediction.

- **SNNS**. Stuttgarter Neural Network Simulator [10] is probably the largest available free tool on neural networks. It implements a large number of neural network models and learning algorithms. SNNS is operated via a graphical interface and a vast number of methods and parameters. However, a user needs some time and experience to master this tool. SNNS is well suited for research and development, but less suitable for problem-oriented industrial users.
- **Data Manager**. This software provides the following functions:
  - selection of a specific day from the database;
  - constructing of a pattern by giving the number of inputs and outputs;
  - aggregating five minute values into 10 or 15 minute values;
  - smoothing three subsequent values;
  - normalising the input values for entering into the simulator;
  - denormalising the output values for visualisation of the results.

4. Conclusion

In this study, we have considered three types of forecasting: weekly, daily and hourly prediction as long-term, mid-term and short-time predictions respectively. The weekly prediction relies on the historic pattern stored in the knowledge base, while for daily prediction, for each day of the week and for each predicted event, a reference pattern is stored which can be used for corresponding predictions. The hourly prediction, however, is a short time prediction which gives a view of the current traffic situation.

We have analysed the potential of neural networks for the prediction of road traffic. For this purpose, a neural network based on a Jordan architecture was designed and trained using backpropagation techniques to predict the future values of the traffic time series using its past values. The trained values were compared with the corresponding actual values, and they were found to be in close agreement.

In our application we found that the MSE is less than 0.003. These results are better than the compared methods. This model improves the forecasting by about 20% for the road traffic flow.

The neural network method requires a database to perform the learning task successfully. Therefore, the database has to be updated frequently. Effectively, with a single neural network it is possible to forecast the traffic ahead with a relatively small error using time series. It is nevertheless possible to obtain a good representation of the traffic. The method presented could be used for other applications such as meteorology and energy.

4.1. Related Research

In the literature, some traffic management systems have been reported. The following is not a complete list of developed systems, however it covers the systems using approaches which resemble the work represented in this paper, and hence provide a sound basis for comparison.

One approach that handles this problem is shown by Engels and Chdnas [11], and is based on a mathematical model. Data is aggregated to 15 minute intervals, and is processed with the discrete Fourier transform. The representation of data in the frequency domain provides the Fourier coefficients. The mathematical function of the pattern is described as a sum of waves with different periods. However, for a dynamic process the information is very often partial and incomplete, and the prediction cannot be based on mathematical models.

The ARIAM system automatically provides traffic

and congestion information collected from more than 1600 loop detectors to radio broadcasters [12]. It includes a traffic data support system with the improved incident detection and prediction models, as well as dynamic application of data analysis techniques such as fuzzy classification and frame-based reasoning. The system allows the traffic authority not only to monitor the traffic state at the detection sites, but also to get real-time, section-related information for all monitoring sections, and additionally on all relevant intersections, weak points and sub-networks. The system makes an extensive data analysis which results in a very complex system.

Another traffic management system which is already in operation has been described by Krause and Altrock [13]. Here the road surface and fog conditions extracted from environmental data are evaluated using a fuzzy logic approach. It delivers more reliable results for a huge range of weather conditions. The experience in using a complex fuzzy logic model has shown that fuzzy logic is suitable to evaluate and analyse multiple interrelating data. However, the number of variables used reached 58! The number of rules used in this model is 1060!

It is important to note that the chief innovation of our approach lies in the fact that we use only one variable! We are convinced that this parameter implicitly inherits other relevant parameters. This is our strong presumption at the present time, and we hope we will be able to prove this for real conditions.

4.2. Further Research

We have not studied the approach in real time conditions to know how the neural network can respond to *unpredicted events* such as accidents. This is a task for our future research.

At present, the final development of the predictor is in progress. The objective is to improve the forecasting capabilities for more than one day. This represents a challenging but very attractive target.

Acknowledgement. The author thanks L.I. Porges for his technical help and for helpful discussions.

References

1. Box GEP, Jenkins GM. Time Series Analysis Forcasting and Control. Holden Day, 1970
2. Weigend AS, Gershenfeld NA. Time Series Prediction: Forcasting the future and understanding the past, Addison-Wesley, 1994
3. Jordan MI. Serial order: A parallel distributed processing. In JL Elman and DE Rumelhart, editors, Advance in Connectionist Theory: Speech, Elbaum, 1989
4. Warg S. Verkehrsstrom-analyse und -prognose mit fuzzy pattern klassifikation. Technical report, Diplomarbeit, TU Chemnitz, Lehrstuhl für Systemtheorie, 1997
5. Porges LI. Zeitreihenmodellierung und prognose mittels fuzzy pattern klassifikation. Technical report, Diplomarbeit, TU Chemnitz, Lehrstuhl für Systemtheorie, 1996
6. Hertz J, Krogh A, Palmer G. Introduction to the Theory of Neural Computation. Addison-Wesley, 1991
7. Williams RJ, Zipser D. Gradient-based learning algorithm for recurrent networks. In Y Chauven and DE Rumelhart, editors, Backpropagation: Theory, Architectures and Applications, Erbraum, 1995
8. Elman JL. Finding structure in time. Cognitive Science 1990; 14
9. Mozer MC. Neural net architectures for temporal sequence processing. In AS Weigand and NA Gershenfeld, editors, Forcasting the Future and Understanding the Past, Addison Wesley, 1994
10. Zell A. Stuttgarter neural network simulator. Technical report, University of Stuttgart, 1995
11. Engels J, Chdnas C. Learning historic traffic applications as basis for traffic predication. In HJ Zimmermann, editor, Proc 5th European Congress on Intelligent Techniques and Soft Computing, EUFIT'97, Vol 3, Verlag Mainz, 1997
12. Kirschfink H, Lange R, Weber R. Data analysis applications for traffic control and monitoring applied in the monitoring network of hessen. In HJ Zimmermann, editor, Proc 5th European Congress on Intelligent Techniques and Soft Computing, EUFIT'97, Vol 3, Verlag Mainz, 1997
13. Krause B, Altrock C. Fuzzy logic analysis of environmental data for traffic control. In HJ Zimmermann, editor, Proc 5th European Congress on Intelligent Techniques and Soft Computing, EUFIT'97, Vol 2, Verlag Mainz, 1997