CS289A Course Project

DRAFT: AN ARTICLE CREATED USING LaTeX2ε IN ASME FORMAT

Harry H. Cheng
Integration Engineering Laboratory
Department of Mechanical and Aeronautical Engineering
University of California
Davis, California 95616
Email: hhcheng@ucdavis.edu

First Coauthor
Second Coauthor
Department or Division Name
Company or College Name
City, State (spelled out), Zip Code
Country (only if not U.S.)
Email address (if available)

ABSTRACT

This article illustrates preparation of ASME paper using LaTeX2ε. An abstract for an ASME paper should be less than 150 words and is normally in italics.

1 Introduction

Macroscopic freeway traffic models are widely used by transportation engineers in order to design controllers and perform re-routing or road management to improve performance of the system; for instance, increasing total traveled distance and decreasing total travel time of vehicles in the network. A crucial component of this process of traffic management is simulation of traffic behavior which is usually accomplished using traditional simulators where system's behavior is predicted by the dynamics assumed to govern evolution of traffic.

The burdensome aspect of traffic prediction is that traffic condition is contingent on too many different parameters. For instance, for a given network, the day for which the simulation is required, plays a significant role as traffic regime on weekends varies from the one on workdays. Furthermore, for a particular day, clearly, vehicular traffic in evening differs from the one at midnight. Thus, a decent traffic prediction must incorporate distinct parameters influencing traffic conditions.

As every watchful motorist driving in California freeways can recognize, there are numerous implemented loop detectors in freeway stretches of California that measure and record traffic metrics. These sensors provide traffic data-centers with huge amount of information that must be taken into account while making predictions. The immense existing data and complexity of traffic behavior as a result of being affected by different continuous and discrete variables, motivated the authors to perform traffic prediction deploying neural network and deep learning methods to come up with a traffic simulation which can possibly perform better than the existing traffic simulators [6].

1.1 Fundamental Diagrams

In transportation engineering, most of macroscopic freeway traffic models take advantage of the assumption of existence of a *fundamental diagram* for each link in the freeway which maps observed vehicular flow in the link to the corresponding vehicular density [5]. Generally, in order to obtain the fundamental diagram for a given link, vehicular density and flow are recorded for various traffic conditions and a curve is fit into the observed data [1]. In first order freeway traffic models which are widely used in controller design procedure, a piece-wise linear function is fit to the data as shown in figure 1. Consideration of a piece-wise linear function implies that flow in a link is restricted to be in three different conditions; *free-flow* region which is the left side of the triangular shape with a positive slope (v_i), flow being equal to *capacity*(F_i) which is the peak point of piece-wise linear function and *congested* region corresponding to the right side of the triangular shape with negative slope (w_i).

1.2 Cell Transmission Model

The most prevalent first-order freeway model presumed to rule dynamics of freeway and extensively in use is Cell Transmission Model [4] in which freeway dynamics are captured by density and flow of each link assuming that the link has a trian-

gular fundamental diagram. In CTM, a freeway is divided into N links and discrete time intervals are considered. For a given link i at time k, the state of the link (density) is updated by equation 1 where $n_i(k)$ is the density of vehicles in section i at time k, $f_i(k)$ is the flow leaving section i at time k and $f_{i-1}(k)$ is the flow entering section i from section $i-1$.

$$n_i(k+1) = n_i(k) - f_i(k) + f_{i-1}(k) \tag{1}$$

As stated already, the assumption is that for each link, fundamental diagram is known; thus, for each link flow is evaluated by equation 2 where $\bar{n}$ is the jam density or maximum number of vehicles a link can accommodate. The *min* term in equation 2 suggests that freeway dynamics is dictated by a non-linear law. Equations 1 and 2 also indicate that each link is coupled to the adjacent links through the flows entering and exiting links.

$$f_i(k) = \min\left(v_i n_i(k), F_i(k), w_{i+1}(\bar{n}_{i+1} - n_{i+1}(k))\right) \tag{2}$$

1.3 Fundamental Diagram Calibration

The aforementioned coupling and non-linearity insinuates that calibration of fundamental diagram for each link must be performed such that traffic behavior is consistent all over the network; in other words, calibrated fundamental diagram of a link may not be accurate enough by considering isolated flow-density relation of the corresponding link. On the other hand, parameters of a fundamental diagram vary over time since traffic regime alters during the day. In order for a simulation by a macroscopic model to represent traffic behavior, the assigned fundamental diagrams are tuned so as to find the set of fundamental diagrams yielding to the observed traffic data which is of course a cumbersome process. Furthermore, due to the time-varying nature of traffic, the whole process should be repeated for different time intervals during which traffic parameters are supposed to be constant [5].

1.4 Inspiration

The aforestated complexities of conventional macroscopic simulators led the authors to exploit the massive amount of traffic data to go beyond piece-wise fundamental diagrams and piecewise affine dynamics of CTM and perceive a multi-input multi-output function in which time of the day and day of week are taken into account as inputs as well. Clearly, the dynamics and evolution of traffic variables is captured by the recorded data regardless of whether there exists a triangular fundamental diagram dictating flow-density relations for a given link or not, revealing the fact that there is no need for burdensome process of calibration. Moreover, simulation performance and accuracy can perhaps improve; for, extra information such as date and time are provided to the trained multi-layer neural network model [9].

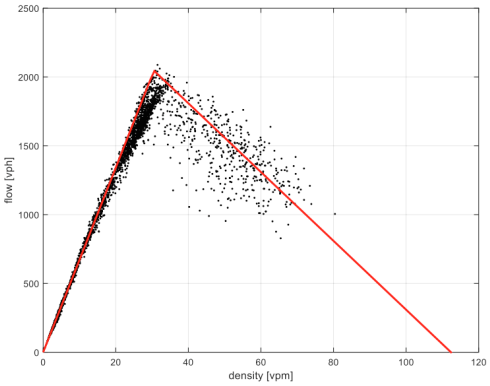

FIGURE 1. Fundamental Diagram of a Link in I-210 east

2 Data Analysis

2.1 PeMS Data

PeMS (Performance Measurement System) is a traffic metrics measurement system all over California deployed to obtain loop detector data in real time [2]. Freeway PeMS data consists of measurements of vehicular flow, occupancy (fraction of time a vehicle is present on the loop [10]) and speed in time intervals of 5 minutes. PeMS data is available to users at `http://pems.dot.ca.gov/`. In this project, I-210 east (Figure 2) PeMS data was used as the training and validation data set.

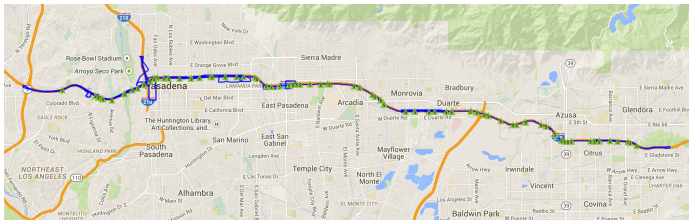

FIGURE 2. I-210 East Topology

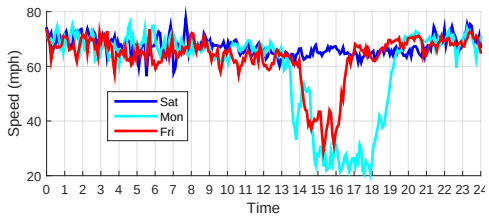

FIGURE 3. Average speed for three days.

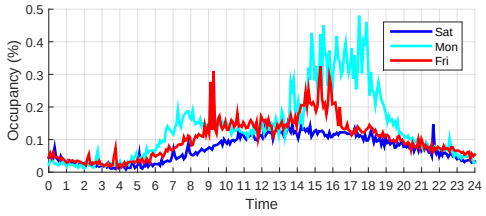

FIGURE 4. Average occupancy for three days.

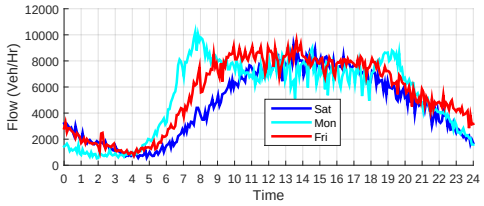

FIGURE 5. Average flow for three days.

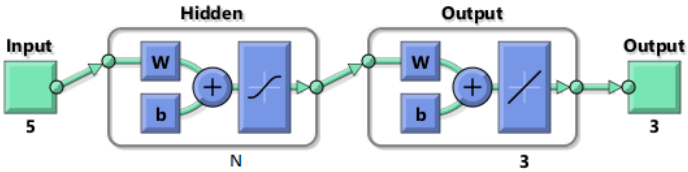

FIGURE 6. Initial neural-network layout with one hidden and one output layer that have *tanh* and *linear* activation functions respectively

2.2 Training Data

As mentioned previously, since traffic behavior is highly dependent on time of the day and date, our training data consists of flow, occupancy, speed, date and time of day for a single VDS (vehicle detection station) of I-210. Each day is divided into 288 intervals where each time interval lasts 5 minutes. This data was collected for the last three months of 2014 (October 1st to December 31st).

2.3 Training Method

Our main focus of data training was obtaining one-step prediction of traffic behavior; in other words, giving the features of a particular time step of a specific day as the inputs, the outputs should be flow, speed and occupancy of the considered VDS in the next time step. In order to get such a prediction function, a multi-layer neural network was trained on the data, whose characteristics such as number of hidden nodes and number of hidden layers were obtained by cross-validation over a range of node numbers and number of layers.

2.3.1 Neural Network Layout It is known by Universal Approximation Theorem [3] that any continuous function on a bounded domain can be approximated by a two-layer neural network with a finite number of hidden nodes. Thereby, the very first step in implementing a NN (deep) learning algorithm is to start with a neural network that has one hidden layer and one output layer as shown in Fig. 6.

In such an architecture, there are three crucial design parameters, namely the number of nodes in the hidden layer, the activation functions for the nodes and the cost function that is desired to be minimized when the trained network is applied to test data. In our running example, the number of nodes in the output layer is 3 since the output vector contains the measurements of *flow*, *occupancy* and *speed*. The input vector, here, contains the same signals with one step delay as well as the corresponding time and day number (i.e. a vector in $\mathbb{R}^5$).

As mentioned above, a design parameter determining structure of the network is number of hidden nodes, denoted by N in Fig. 6, that should be determined prior to the training process. In order to decide on the "optimal" number of hidden nodes, a range of values from 2 to 80 was chosen. We considered *tanh* as the activation function for the neurons in the hidden layer, and a pure *linear* function for the output layer nodes (c.f. Fig. 6). The available data was divided to training, validation and test data sets. These sets, respectively, held 70%, 15% and 15% of the full data set.

There are many different algorithms such as "Levenberg-Marquardt" and "Scaled Conjugate Gradient" that can be used for training neural networks. Choosing the right algorithm may be a tedious process since it depends on many factors, including the complexity of the problem, the number of data points in the training set, the number of weights and biases in the network, the error goal, and whether the network is being used for pattern recognition (discriminant analysis) or function approximation (regression). Here, we started off with Levenberg-Marquardt backpropagation [7] since it is often one of the fastest backpropagation algorithms for regression, and it is usually recommended as a first-choice supervised algorithm. Like the quasi-Newton methods, the Levenberg-Marquardt algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. We chose MSE (mean squared error) as the optimization cost function. Validation vectors are used to stop training early if the network performance on the validation vectors fails to improve or remains the same for max-fail epochs in a row. Test vectors are used as a further check that the network is generalizing well, but do not have any effect on training. In MATLAB, training stops when any of these conditions occurs:

1. The maximum number of epochs (repetitions) is reached.
2. The maximum amount of time is exceeded.
3. Performance is minimized to the goal.
4. The performance gradient falls below min-grad.
5. μ exceeds $\mu_{\max}$.
6. Validation performance has increased more than max-fail

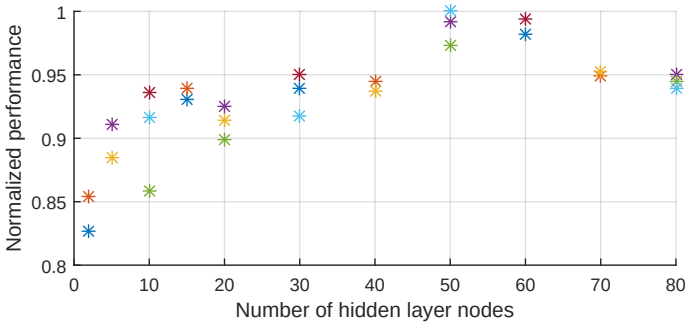

FIGURE 7. The performance of trained neural network being normalized by the maximum achieved performance versus the number of nodes in the hidden layer when *tanh* and *linear* activation functions are used.

times since the last time it decreased (when using validation).

Training for each set of parameters is performed with different random initializations to avoid being trapped in local minima and the the performance of trained network is shown in Fig 7. This figure indicates that the best performance is achieved by approximately 50 nodes in the hidden layer when the activation function is *tanh*. Larger number of nodes in this particular case results in a smaller training error, but at the same time causes over-fitting. The number of network parameters being learned by this network is 453 ($[5+1]\times 50+[50+1]\times 3$). We also considered other activation functions including radial basis and sigmoid and repeated the process with the same design parameters (i.e. architecture and algorithm) and it was realized that among all *tanh* performed better. Moreover, "Scaled Conjugate Gradient" (conjugate gradient backpropagation with Polak-Ribire updates) [8] and "Variable Learning Rate Backpropagation" (gradient descent with momentum and adaptive learning rate backpropagation) were considered as alternative algorithms. Among these three algorithms, Levenberg-Marquardt had the fastest convergence rate and smallest mean squared error. The set of all designing parameters used in the "best" achieved training are listed in Table 1. The code was being run in MATLAB on a machine with Windows Server 2012, Intel(R) Core(TM) i7-3770 CPU and 12.0 GB of RAM. Parallel computing toolbox of MATLAB was used to utilize 4 workers in a parallel pool for the training process.

The next step is determination of number of layers in the desired network. The question of what is the best of way of configuring a particular number of nodes (in this case 50) should be addressed. Evidently, best in the optimal sense is obtained by optimizing over the whole set of possible configurations of the network which is practically impossible; hence, performance of several configurations of the network with 50 nodes is compared (figure **??**) where the best performance is obtained by x number

TABLE 1. Final parameters used for neural network training

Algorithm	Levenberg-Marquardt
Number of layers / parameters	2 / 453
MSE (normalized input/output)	0.051
Number of epochs (time)	117 (88 seconds)

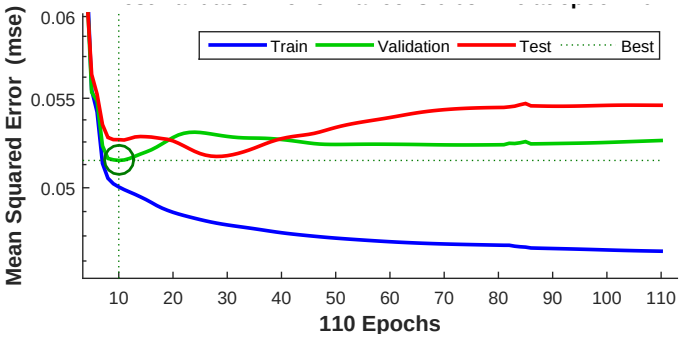

FIGURE 8. Training, validation and test performance.

of layers where first layer has x number of nodes...

REFERENCES

[1] Michael J Cassidy, Kitae Jang, and Carlos F Daganzo. Macroscopic fundamental diagrams for freeway networks. *Transportation Research Record: Journal of the Transportation Research Board*, 2260(1):8–15, 2011.

[2] Tom Choe, Alexander Skabardonis, and Pravin Varaiya. Freeway performance measurement system: operational analysis tool. *Transportation Research Record: Journal of the Transportation Research Board*, 1811(1):67–75, 2002.

[3] Balázs Csanád Csáji. Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24, 2001.

[4] Carlos F Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287, 1994.

[5] Gunes Dervisoglu, Gabriel Gomes, Jaimyoung Kwon, Roberto Horowitz, and Pravin Varaiya. Automatic calibration of the fundamental diagram and empirical observations on capacity. In *Transportation Research Board 88th Annual Meeting*, volume 15, 2009.

[6] Mark Dougherty. A review of neural networks applied to transport. *Transportation Research Part C: Emerging Technologies*, 3(4):247–260, 1995.

[7] Martin T Hagan and Mohammad B Menhaj. Training feedforward networks with the marquardt algorithm. *Neural*

Networks, IEEE Transactions on, 5(6):989–993, 1994.

[8] Martin Fodslette Møller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*, 6(4):525–533, 1993.

[9] Byungkyu Park, Carroll J Messer, and Thomas Urbanik II. Short-term freeway traffic volume forecasting using radial basis function neural network. *Transportation Research Record: Journal of the Transportation Research Board*, 1651(1):39–47, 1998.

[10] Yinhai Wang and Nancy L Nihan. Freeway traffic speed estimation with single-loop outputs. *Transportation Research Record: Journal of the Transportation Research Board*, 1727(1):120–126, 2000.

Appendix A: Head of First Appendix

Avoid Appendices if possible.