Performance of Deep Neural Networks in the Analysis of Vehicle Traffic Volume

Adrian Bosire[1], George Okeyo[2], and Wilson Cheruiyot[3]

[1]*Department of Computer Science, Kiriri Womens University of Science and Technology, Kenya.*
[2 & 3] *School of Computer Science and Information Technology, Jomo Kenyatta University of Agriculture and Technology, Kenya*

***Abstract*– The major problem of vehicle traffic congestions is the increased time wasted in the queues and the resultant high cost of resources used during the same period. Therefore, this research seeks to evaluate the viability of Deep Neural Networks in the performance analysis of vehicle traffic volume. This will assist in effective and efficient traffic monitoring, travel-time forecasting and traffic management. Deep Neural Networks (DNN) offer an optimal option for alleviating the problem of traffic congestion. Although Artificial Neural Networks (ANN) usually encounter setbacks such as local optimum thereby resulting in short term forecasting this can be effectively overcome by using an appropriate training algorithm with correctly configured parameters for the kind of data under consideration. The data is divided into samples for training, validation and testing, after which the overall performance is evaluated using the Mean Squared Error (MSE). The results obtained will help in the evaluation of the practicability of using DNNs in analyzing vehicle traffic flow. Eventually, this can be leveraged for time-forecasting of traffic conditions and also mitigate traffic build-up.**

***Keywords* – Artificial Intelligence, Deep Neural Network, Performance analysis, Traffic volume.**

I. INTRODUCTION

Vehicle traffic congestion is a nuisance to the society with extended effects both environmentally and socially. The domain of vehicle transport presents challenges such as the increased travel time and consequent high cost of resources incurred at such a time due to the poor traffic queue management. The changing factors in this dynamic environment as represented in variables such as the weather, accidents or incidents and even unscheduled events may result into unforeseen traffic snarl-ups which may limit us to short term forecasting which should not be the case. Essentially, we should have a reliable means to forecast with due consideration to such factors that may inhibit the prediction accuracy in order to attain a traffic queue management model which sustains the dynamic nature of the environment.

The performance of the traffic system is an abstract concept which can be described by a number of metrics such as traffic volume, speed, delay, travel time, queue length among others.Moreover, the traffic referred to in this research basically focuses on all the vehicles which use the main roads in the urban areas, where most congestion is experienced. The traffic analyzed will also be restricted to daytime since it is assumed there is slight traffic during the night. Therefore, analysis will aim at increasing the performance of the road networks in the urban environments.

Urban mobility impacts urban life to a great extent. People, living in cities, plan their daily schedule around anticipated traffic patterns [2]. The main factors that commuters used to describe the service level included the traffic volume, the wait time at various traffic signals, road surface and travel conditions, vehicle fuel consumption and the distance covered in case of detours [20.Forecasting and efficient alleviation of road traffic on a real-time basis has become an essential and important element in management of transportation systems [26]. Furthermore, if we can estimate the future location of cars which are already on the road network, we will be able to estimate future congestions and upcoming traffic hazards. These predictions are useful in areas such as congestion prediction, traffic control, upcoming traffic hazards and targeting advertisements next to the roads among others [27].

In a study of over 50 developed and developing world cities revealed that most people strive for cleaner, less congested cities and improved traffic flow [9]. Therefore, these beckons for ideal solutions which are relevant in travel time prediction and equally applicable in spontaneous circumstances. Some of the many schemes or approaches that formulate the basis for travel time prediction have been surveyed and the existing methodologies have been categorized into five commonly used prediction models:historical data-based models, statistical models, Kalman filtering model, machine learning models, and hybrid models [22]. Even so, there are a variety of classifications for time prediction models which were introduced in other studies. For instance, [29] grouped them into four categories: regression method, time series estimation method, hybrid of data fusion or combinative model and Artificial Intelligence methods like Neural Network. Reference [28] also characterized the methods into categories such as historic approaches, machine learning techniques, model-based approaches and statistical methods. Here, the machine learning techniques included Artificial Neural Network and Support Vector Machines, whereas, Kalman Filtering was considered a model-based approach. Statistical methods include regression analysis and time-series.

Subsequently, the fore mentioned techniques can simply be classified as historical (statistical) models, hybrid models and

machine learning (Artificial Intelligence) models. Notably, Artificial Intelligence (AI) prevails in most classifications or is at least used as one of the criteria for deriving the categories. Reference [14] demonstrated that the previously mentioned models excluding the ones inclined to Artificial Intelligence lack the flexibility to deal with non-linear features often present in traffic data. Artificial Neural Network models and Support Vector Regression models address this problem in a principled manner and have gained recent popularity in time prediction because of their ability to deal with complex and non-linear relationships between variables.

This paper is organized as follows: Section 2 presents related studies with an overview of the classical and heuristic techniques used in traffic management. This also includes other cases where neural networks have been used in traffic management. Section 3 presents the methodology used to carry out this study followed by section 4 which covers the experimental setup. This is later followed by the discussion of the results obtained and the conclusion in sections 5 and 6 respectively.

II. RELATED STUDIES

This section provides an overview of related research on the variations in urban traffic volumes. These variations can be analyzed using various techniques and on different time scales, ranging from daily to annual. Therefore, different combinations of time scale and traffic volume aggregation level can be distinguished. We examine Neural Networks and their practicability in enhancing vehicle traffic flow via predictive analysis.

A. Techniques Used In Traffic Management

The terminology "traffic management" has been used to refer to all forms of monitoring, controlling, forecasting and other functionality related to handling of vehicular traffic flow. Most of the techniques applied in short-term traffic flow prediction can be classified broadly into two major groups [17]:

1) Classical Methods which are based on Statistical/ Mathematical concepts and
2) Modern Heuristic Methods which are based on Artificial Intelligence algorithms.

1) CLASSICAL METHODS

Classical methods for time series prediction can be categorized into Exponential Smoothing methods, Regression methods, Autoregressive Integrated Moving Average (ARIMA) methods, Threshold methods and hybrids thereof [17].

Exponential Smoothing schemes obtain a prediction by assigning exponentially decreasing weights to recent observations in the time series. The exponential smoother responds to process changes faster by giving geometrically decreasing weights to the past observations. This means that recent observations are assigned more weight compared to the former and smoothing is achieved by relating the current observation to the previous ones. Simply, consider that a data set is comprised of a signal and noise. The signal indicates any pattern caused by the intrinsic dynamics of the process from which the data are collected. Therefore, Smoothing can be seen as a method aimed at separating the signal and the noise, hence, the smoother acts as a filter to approximate the signal [5].

Regression models presents a prediction based on a linear function of one or more independent variables hence deal with non-linear time series by logarithmic or power transformation of the data, however this technique does not account for asymmetric cycles and outliers. These models seek to investigate the relationship between a dependent variable and independent variable(s). The independent variables are the predictive factors that influence the characteristics of the target variable which is the dependent variable.

ARIMA models make a forecast based on past observations that formulate a linear function. The model uses a combination of factors such as transformational differences that occur in past observations and residual errors caused by some independent variables in the past along the time series. The linear function transforms the input into an output, based on all of the values from the past along with other parameters under consideration. All these values of the input from the past are allocated different weights then represented as a summation.

Threshold methods rely on the assumption that the spikes or changes in a given time series are caused by the values of the independent variables exceeding a certain limit. This means that the various sections are modeled as an exponential or logistic function to study the extents that necessitate either a smooth or abrupt change in behavior. The changes inform a regime switch which may further signify a characterization of different environments in the same domain.

Generally, these classical methods assume that the underlying data generating process of the time series is constant. This is often invalidated in dynamic environments such as in the case of vehicle traffic flow which is usually faced with changing factors that may be scheduled or even unscheduled such as natural events.

2) HEURISTIC METHODS

Modern heuristic methods can be categorized into Neural Networks (NN) based models and Evolutionary Computation based methods.

Evolutionary Computation methods are generally based on evolution thus implement random variation, reproduction and selection by altering and moving data within a computer. In simple terms, the Darwinian principles of natural selection form the basis for implementing and assessing algorithms associated with these methods. The algorithms in this class include Genetic Algorithms, Evolutionary Programming, Evolution Strategies and Genetic Programming [17], [19].

Neural Networks (NN) originally refers to a network of biological neurons. More broadly, the term evokes a particular

paradigm for understanding brain function, in which neurons are the essential computational units, and computation is explained in terms of network interactions. Although neurons are biological entities, the term Neural Network has come to be used as a shorthand for Artificial Neural Network, a class of models of parallel information processing that is inspired by biological neural networks but commits to several further major simplifications [18].

So, Neural Networks are mathematical models that try to emulate some functionalities of human brain, particularly in relation to the learning mechanisms. They are based on the concept that it is possible to train a mathematical model so that it is able to reproduce some physical phenomena or to forecast some results; the model is trained with samples, that are the actual results (outputs) of the studied system corresponding to the actual values (inputs) of some variables of the problem. Generally, the training phase requires sampling a high number of real cases [7].

Reference [7] classify Neural Networks into three different classes based on their network architecture:

1) Single-Layer Perceptron Networks
2) Multilayer Perceptron Networks
3) Recurrent Perceptron Networks

The Single-Layer Feed-Forward Networks are composed of one input layer of source nodes connected to one output layer of neurons as shown in fig.1 below.

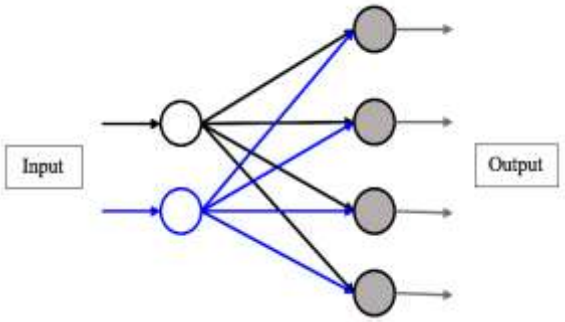

FIGURE 1. Single layer perceptron model

The Multilayer feed-forward network is the most developed and implemented. The main feature of this model is the existence of hidden layers consisting of hidden neurons (perceptrons) that perform the actual computations. These hidden neurons form a computational connection from the external inputs to the network outputs. The nodes at the network input layer feed the input vector to the corresponding elements of the next hidden layer which equally propagates the received computed input signals and applies it to the neurons in other layers until it gets to the output layer. The propagation of the signal is carried forward without cycles and without cross-connections. This is shown in the fig. 2 below.

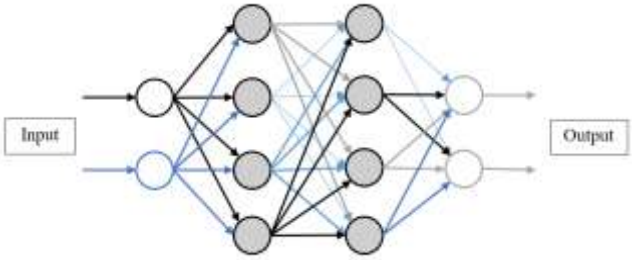

FIGURE 2. Multi-layer perceptron model

A Recurrent neural network has at least one feedback loop. This means that the network has one or more cycles, hence making it possible to follow a path from a unit back to itself. The presence of cycles means that the network remembers everything because all the inputs are related to each other. Such a memory, improves the learning capability of the network eventually having an overall positive impact on the performance. The information that is preserved in the loop forms the memory that serves to keep track of the context. The fig. 3 below illustrates the concept of the recurrent neural network.

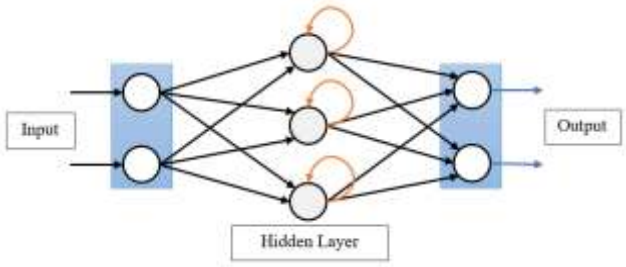

FIGURE 3. Recurrent Neural Network Model

For any kind of Neural Network, the information from a layer to the following one are transmitted by means of "transfer functions". The transfer or activation function makes a calculation that relies on the weighted sum and bias to determine whether a neuron will be activated or not. This calculation introduces non-linearity to the network which helps it to learn (understand the underlying complexity) so as to produce accurate results. Therefore, the activation function is a non-linear transformation performed on the inputs and sent through the network making it capable to learn and perform complex tasks as exhibited in dynamic environments

Many transfer functions can be used in order to fit several different input-output experimental patterns. For example, a simple classification problem may be solved by adopting a Boolean-type transfer function (i.e. a function that produces only O and 1 as outputs); then, according to the increasing complexity of the problem, linear, sigmoid, logarithm, etc. transfer functions can be adopted. The "optimal" transfer function depends on the physical process to be represented; it can be chosen after a sensitivity analysis among different options.

During the learning phase, the model learns how to connect inputs to outputs; it, essentially, defines the shapes of transfer functions. There are different types of learning methods, depending on complexity and structure of the network:

1) Supervised learning;
2) Unsupervised learning;
3) Reinforcement learning.

For solving the learning problem different algorithms can be used. The goal of these algorithms is to find the precise weights that enable the network connections to make accurate decisions in a process known as training. Therefore, we use a cost function as a benchmark our progress in determining the right weights. The learning algorithms basically use the gradient descent of the cost function to gradually adjust the weights to get closer to the optimal values. One of the approach for calculating the gradient is using the backpropagation technique, which computes its value backward through the network; indeed, the term "backpropagation" is related to the technique used to calculate the derivatives of the error function. The Fig. 4 below shows a classification of Neural Networks [30].

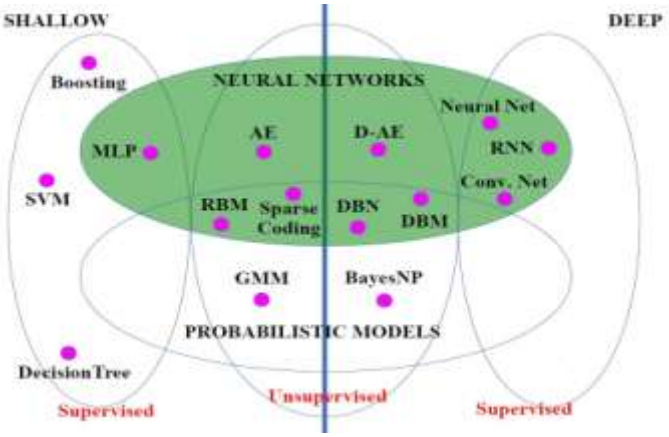

FIGURE 4. Classification of Neural Networks[30]

Therefore, Neural Network based models can broadly be categorized into Shallow Neural Network (SNN) techniques and Deep Neural Network (DNN) techniques. SNNs are Neural Networks that generally have only one hidden layer as opposed to DNNs which have several levels of hidden layers. Reference[3] reformulate the claim that deep networks can solve more complex problems by proving that Deep Neural Networks can instead utilize functions whose complexity is of a higher magnitude contrary to Shallow Neural Networks, given that all resources remain constant.

Shallow Neural Network techniques typically contain one or at most two layers of nonlinear feature transformations. Examples of the SNN techniques are Conditional Random Fields (CRFs), Gaussian Mixture Models (GMMs), Support Vector Machines (SVMs), Maximum Entropy (MaxEnt) models, Logistic Regression, Kernel Regression, Multi-Layer Perceptron's (MLPs) with a single hidden layer including Extreme Learning Machines (ELMs). However, SNN techniques effectively solve well-constrained problems (these are problems for which a function is to be minimized or maximized with respect to well defined constraints) due to their limited modeling and representational power which poses a challenge when dealing with complicated real-world applications [3], [11].

Conversely, dynamic environments suggest the need for Deep Neural Network techniques which can easily be used in extracting complex structure and building internal representation. This concept originated from Artificial Neural Network (ANN) research. Therefore, we define Deep Neural Networks (DNN) as Artificial Neural Networks composed of several interconnected hidden layers, having multiple hidden nodes (perceptrons) for computational use, between the network input layer and its network output. Feed-Forward Neural Networks or MLPs with many hidden layers, which are often referred to as Deep Neural Networks (DNNs), are good examples of such models. Back-propagation (BP) is also a preferred algorithm for training these networks. However, the pervasive presence of challenges such as local optima in the non-convex cost function of the DNNs are the main source of difficulties in the learning process [4], [6].

B. Neural Networks In Traffic Management

Reference [26] performed field tests under real traffic situations in order to test the prediction of bus arrival time to stations depending on two techniques consisting of Kalman Filter and Neural Network. In their experiment, they found it was difficult to determine a good network topology from only the number of inputs and outputs when it comes to Neural Networks. However, some of the crucial factors to consider are the training sample and the complexity incurred during classification. So, for a more accurate decision, it was critical to choose the suitable number of hidden layers for the Neural Network. Ultimately, the Mean Square Error (MSE) decreased with an increase in the number of hidden layers but the processing time also increased. This simply means the accuracy increased but with a parallel increase in complexity. The analysis of field data obtained for 12 stations, based on the Root Mean Square Error (RMSE) of the predicted arrival times and the actual arrival time, indicated that the Neutral Network algorithm outperformed the Kalman Filter in 10 stations. The Kalman Filter algorithm was better in 2 Stations but by small value not more than 0.6 minute. Moreover, even though both algorithms are affected by abrupt changes in traffic volume, the effects were notably higher in Kalman Filter as compared to the Neural Network.

Mathematical spatial - temporal algorithms have also been used as an optimal algorithm for travel time prediction in the urban areas. Specifically, the Kalman Filter and Neural Network algorithms. Both algorithms were tested and assessed on the Mean Square Error and the Mean Absolute Error (MAE) and were found to be cost effective due to the minimal variation obtained. ANN had a Mean Square Error of 3.8308e-11 and the MSE of Kalman Filter was 2.06635273e-7. However, a critical analysis of the statistical variations of the MSE and an average

MAE of 2.77661506021e-4 for Kalman Filter and MAE of 2.5415192e-7 for Artificial Neural Network indicates that the performance of Artificial Neural Network is much better than Kalman's Filter. Then again, ANN is cumbersome at the training phase especially with the selection of input variables and adjustment of weights so as to achieve optimal performance [10].

A simplified stochastic approach which is simply based on the probability density function in travel time prediction has also been used. Furthermore, there are three conditions that useful forecasting systems require. First, they should have a low load factor to the overall operational system. Second, they should have a simple approach to calculate required parameters thus simplifying their management. Third, they should have a high prediction accuracy preferably with an error rate of less than 5%. The forecasting methodology implemented satisfied these conditions and was compared against a historical average method. In the case of forecasting long-distance paths, using historical average method was more accurate than using stochastic method but the stochastic method is more flexible in forecasting when traffic flow condition is unusual, such as holiday seasons, during peak hours and generally when there was a high fluctuation rate in traffic flow. Results based on standard deviation assessment showed higher prediction accuracy during peak time thereby having a bias for higher density or higher traffic volumes and showed reliability when the traffic was fluid. These partialities show a lapse in reliability and robustness which should not be the case in real-time systems [23].

The use of ANN incorporated with a data fusion technique was also proposed and implemented. The proposed method uses Dezert-Smarandache Theory (DSmT) as a fusion technique and Artificial Neural Network (ANN) as mining tool. The Dezert-Smarandache Theory (DSmT) is a natural extension of the classical Dempster-Shafer Theory (DST). DSmT allows combination of any type of independent sources of information which is highly conflicting and imprecise as well as quantitative or qualitative sources of evidence. Therefore, the basic idea of data fusion is to estimate parameters by using more than one measurement from different sources or sensors. There are different methods to fuse data ranging from a sample arithmetic mean to a more complex DF approach however, they can be classified into three major categories; Statistical approaches, Probabilistic approaches and Artificial Intelligence (AI) approaches. Neural Networks fall under the AI approaches. Nonetheless, the model estimates are corroborated using actual values and the results show the model performed well and produced acceptable prediction based on the performance measures that were used; Mean Absolute Error (MAE) which was 2.5776, Mean Absolute Percentage Error (MAPE) at 6.8614% and the Root Mean Square Error (RMSE) at 3.4233. These values were well within the threshold hence, were sufficient for the assessment. The acceptable performance in travel time estimation was owed to the minimal differences between the actual times realized and the estimations as predicted. However, the study was restricted to short distances among other challenges such as difficulty in finding the effective implementation of the combining rule, difficulty in applying the rule on filtered parameters and more effort is required in resolving the conflicts thereof [16].

ANN models have gained popularity in time series prediction because of their ability to deal with complex and nonlinear relationships between variables. Even though these models are preferred they suffer from slow learning process and are difficult to interpret and implement. However, they model travel times using Additive models which provide a principled statistical framework for arrival time predictions. The main advantage of Additive models in this context of forecasting is their ease of interpretability and flexibility in modeling complex non-linear relationships. In particular, they model cumulative travel time as a smooth function of route location and further allow this functional relationship to vary smoothly across time. The overall performance of the Additive Mixed Model (AMM) compared to the Basic Additive Model (BAM), Extended Additive Model (EAM), Kernel Regression and Support Vector Machine (SVM), was evaluated based on the calculated Mean Absolute Relative Error given as a percentage. On average the AMM model outperformed the rest by 1%. Besides, these Additive models are disadvantaged since they tend to put more priority on minimizing error, when it is in fact larger, at the expense of short-term predictions [14].

Reference [27] investigated the application of Neural Networks to time series prediction or forecasting. Several points are to be addressed when investigating such applications, including the selection of an appropriate approach for the Neural Network, finding the appropriate representation of data and subsequent coding of the algorithm as well as overall implementation. During the selection of the approach, one can opt to use either design; Feed-Forward or Recurrent. In Recurrent Networks, as opposed to Feed-Forward Network, neurons are allowed to have connections to outputs of neurons from the following layers. However, it should be noted that Recurrent Networks tend to be significantly more difficult to train although they may be well suited to time series prediction due to their knowledge of previous states. Nevertheless, Feed-Forward Network approach is the more widely used but we must provide for time delay of input signals explicitly. The other factors of coding are dependent on personal preference and prowess in a given programming language as well as the software and hardware resources available for use.

ANNs are based upon biological neural networks by mimicking their design and information processing in a simplified manner. They both consist of processing elements called neurons that are highly interconnected making the networks parallel information processing systems. They are capable of tasks such as pattern recognition, perception and motor control that are considered poorly performed by conventional linear processing. These parallel systems are also known to be robust and to have the capability to capture highly non-linear mappings between input and output layer. On the other hand, imitating natural phenomenon presents a challenge

since we can never fully replicate a concept without embracing the shortfalls that come along with the approach [1], [8], [22], [31].

Nevertheless, these ANNs are prone to some setbacks especially in training them so as to better predict or forecast [25]. Some of these problems which include the non-convex problem, curse of dimensionality, vanishing gradient and hyper-parameter value selection among others. He goes a step further and suggests some time-tested stopgaps which imply that Deep Neural Networks are superior to the Shallow Neural Networks. However, DNNs come at a cost of time and computation but some of their features in terms of functionality offer solutions to the fore mentioned drawbacks. Some of the solutions include the Backpropagation (BP) algorithm and use of stochastic gradient. They simplify the computation as well as reduce the time cost. Therefore, it is desirable to choose the Deep Neural Network which is basically an advancement of the comparatively Shallow Neural Networks so as to overcome some of the problems, of course at a cost of time and complexity which can be resolved via optimal configuration of selected parameters [15].

III. METHODOLOGY

In order to overcome the issue of traffic snarl-ups and the problem of local optima experienced by the Deep Neural Networks. First, the model is conceived and developed and then the dataset is examined for consistency and eventually evaluated using the model. Finally, the results are compared against the real traffic dataset for a certain period which serves as a benchmark for the performance of the model in terms of traffic forecasting.

Nowadays logistics management and transportation networks are growing in both size and complexity. This increase is driven by the higher demand of quality service levels with noticeable efficiency. This in turn complicates the type of data that is managed and utilized by such systems [21]:

The dataset will comprise of various attributes filtered from some of the sources such as weather data, social media, scheduled events and traffic data from simulations as well as traffic gantries mounted along the roads. These attributes will form the variables that will be evaluated at the input layer of the Deep Neural Network selected. The data will be filtered in terms of the time and validity. The dataset will be presented in terms of Comma Separated Values (CSV) format so as to standardize it from the various information sources aforementioned.

The selection of variables for input from the attributes available from the dataset is essential to ensure consistency. Therefore, the validity of the data should be assessed in a time-sensitive manner. The dataset also needs to conform to the standard CSV format. Once the same has been achieved then the dataset is ready for processing. The most important division of the dataset is the one that is going to be used in the training phase and validation phase, then follows the actual testing phase dataset that is going to be used for the comparison and overall evaluation of the model [24].

The data collected then has to be stored and processed. Data from various detector stations is structured in such a way that useful items can be extracted easily and stored preferably in a format such as an Excel sheet or Comma Separated Values. The data record should include a time stamp and a location code. These codes enable the data to be combined with other time and/or location specific information. Other erroneous and inaccurate data due to discrepancies by the data collection system are considered invalid and in such a case either it is discarded or simulated data based on historical occurrences is used in its place. Nonetheless, such instances will explicitly be noted.

As for the specifics, Recurrent Neural Networks will be used. This DNN is preferable because of its architecture which allows for all inputs to correlate to the various input variables identified during the training phase. Implementation of this network produces a disparity between the network output and the target output.This constitutes of an error that will act as the measure used to adjust the weights of the neurons iteratively in subsequent layers of the network so as to minimize the total error of all input/output pairs.

The Neural Network Toolbox for MATLAB provides functions and applications for modeling complex non-linear systems that are not easily modeled with a closed-form equation. The Neural Network Toolbox supports both supervised learning and unsupervised learning. Moreover, the toolbox can be useful for applications such as data fitting, pattern recognition, clustering, time-series prediction, and dynamic system modeling and control [13].

The traffic data can be combined with calendar events data, weather data, data on road works, social media data and accident or incident data in order to explain variations in measured traffic volumes. The data can be used for the analysis of variations in urban traffic volumes.

Mathworks Software Company also offers data visualizing tools such as MATLAB graphics and MATLAB Data Analysis which can easily be integrated with Neural Networks toolbox. By analyzing traffic data insight can be obtained into existent traffic patterns. The results of the cluster analyses can be used to monitor regular and irregular variations in daily traffic flow. This insight provides information on the time and location of bottlenecks in the road network and can be used for traffic planning, land use planning and for taking adequate traffic management measures. The recurrent traffic patterns can be explained for by variations in activity patterns or travel behavior and thus can the results of also be used for traffic monitoring in other cities and towns.

IV. EXPERIMENT SETUP

The design process for the various experiments performed, adhered to the following steps [12].

1) Preparing the dataset
2) Creating the Neural Network
3) Configuring the Neural Network Inputs and Outputs
4) Setting the initial values for the weights and biases
5) Training the Neural Network
6) Validating the Neural Network
7) Evaluating the Neural Network

The first step involves the preparation of the dataset in terms of collection and cleaning through verification and validation to ascertain its viability and validity. Thereafter, a Neural Network is created, and it is then configured and trained.

The configuration phase involves the organization of the network parameters in order to make it compatible with the problem to be solved and also with respect to the dataset. Afterwards, the weights and biases are appropriately adjusted so as to enhance network performance. The process of tuning these adjustable network parameters is referred to as Neural Network training. Remember, both steps of configuring and training the Neural Network require sample data, usually a percentage of the dataset. After setting up the initial values for the weights and biases training of the network can now begin. In fact, most Neural Networks can be trained for data-fitting, time-series prediction or pattern recognition. So, we monitor the progress during the training phase with a keen focus on the performance metrics. We also pay attention to the magnitude of the gradient and the number of validation checks because they are used to terminate the training phase.

The performance of the network during training will decrease to a minimum as the gradient gradually decreases. This performance of the network under observation is measured by the errors obtained during the three phases of training, validation, and testing. However, it is often useful to investigate the network response through a regression analysis. The effected

On termination of the training phase, we verify whether any changes need to be made to the adjustable parameters. Especially, changes to the parameters that affect the network architecture and the datasets may have an overall effect on the network performance. If the network is not sufficiently accurate we initialize the network again with different network parameters and retrain with an aim to produce different solutions. One approach is to vary the number of hidden neurons in the hidden layers. The greater the number of hidden neurons translates to more optimization parameters and computational power which means greater flexibility for the network. Alternatively, we could try a different training function since one may produce better generalization capability than another.

However, the training of the Neural Network usually yields different results. This is caused by the disparity in the adjustable parameters. Each time we run the tests we stand to obtain a solution closer to the optimal or further from the best outcome. Nonetheless, retraining the Neural Network is essential to guarantee the viability and accuracy of the outcomes presented. The overall performance will be evaluated on the various results obtained and a conclusion made courtesy of the various environments created by adjusting the mentioned parameters [12].

V. RESULTS DISCUSSION

Recurrent Neural Networks (RNN) take as their input the current input dataset and also what they have perceived previously in time. This means that part of the output is fed back as input therefore creating a loopback mechanism on each hidden layer. For this experiment, the dataset is logically split into three sets; 70% is used for training and the other 30% is equally divided for both validation and testing. This dataset division basically helps with the generalization capability of the Neural Network. Principally, this can be achieved through either of the two main techniques of early stopping or regularization. Early stopping is the technique in which the dataset is divided into three subsets whereas in regularization technique, the cost function is modified in a manner to yield minimal values closer to the optimal goal.

We use computations obtained by the network on the training set to gradually change the source node input weights and bias values. This process produces errors that are squared and summed to yield the cost function. At the training phase the validation set is observed since it is used in ensuring that the network performs generalization and stops its training process before it starts to overfit. Both the training set error and the validation error decrease during the initial phase of training hence this stage should be monitored closely.

However, overfitting of the data is typically signified by a steady rise in the error on the validation set. The consistency in this error will stop the training process, and the adjustable network weights and biases at the minimum of the validation error are considered. We also plot the testing set error during the training process. Therefore, when there is a disparity in both the testing and validation data set errors in regards to getting to the minimum values at different iterations demands a review of the dataset division.

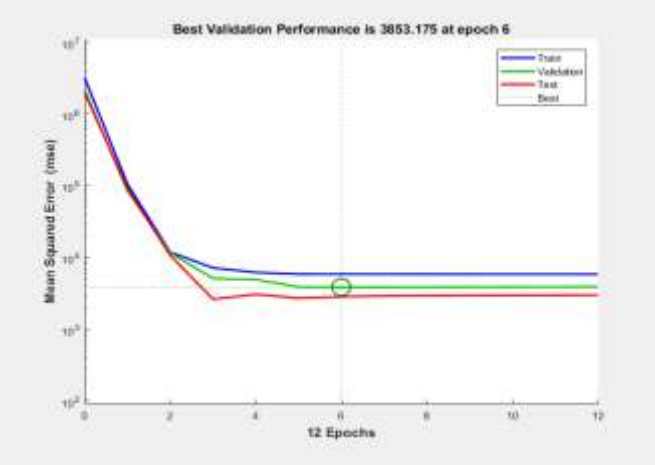

FIGURE 5 Recurrent Neural Network Performance Curve

The plots of the three divisions of the dataset in fig. 5 above depict the performance of the Recurrent Neural Network during the experiment. The Mean Square Error (MSE) is used to measure the performance of the Neural Network.

The MSE is an assessment of the models estimates against the actual test values which creates a difference that on average is treated as the error which is squared to convert the numbers obtained to an absolute value. In this case, after training the Neural network, the validation MSE of 3.853e3is achieved which is relatively low.

The training MSE is comparatively lower as summarized in the table 1.0 later. Lower MSE values are better meaning that the accuracy of the Neural Network is high. In addition, a value of zero indicates no error which ideally indicates a perfect situation.

Next, we create the regression plot as illustrated in fig. 6.

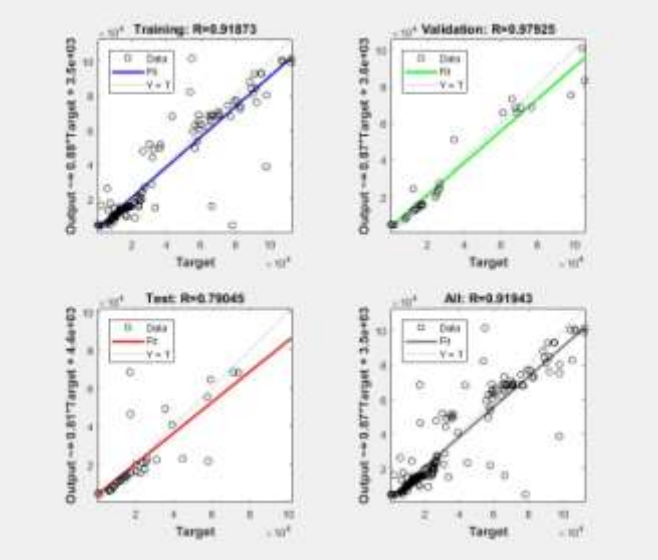

FIGURE 6.Recurrent Neural Network Regression Plots

The regression plot in fig. 6 above shows the relationship between the network outputs and the targets. Essentially, a perfect situation means that training was perfect which in turn means that the network outputs and the targets are exactly equal, but this is hardly the case in practice. Therefore, in the fig. 6 above, the regression values measure the correlation between outputs and targets. Each of the four plots has a dashed line representing the perfect results and the solid line represents the line of best fit obtained from our datasets. The line of best fit is a linear regression line between the generated network outputs and the target values. The value of R at the top of each plot indicates the relationship between the network outputs and targets in the range 0 - 1. If the value of R tends towards 1, it signifies a close linear relationship between the network outputs and the targets but if the value of R tends towards zero it denotes lack of a linear relationship between the generated network outputs and the targets. From fig. 6 above the regression value for the training set, validation set and test set all tend towards one indicating a high correlation between the outputs and the targets. Also, the last plot combines all these datasets and plots the regression to obtain a value of 0.91943 which is very close to 1.

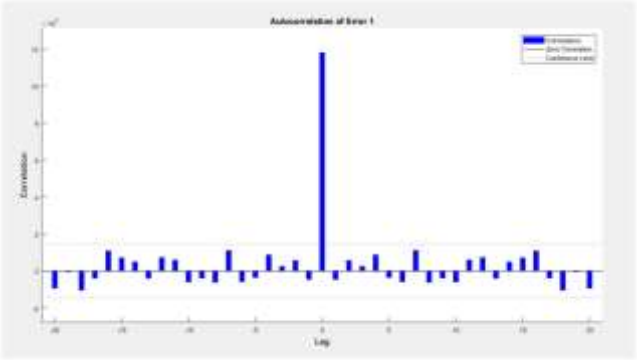

FIGURE 7 RNN Training Error Autocorrelation

The error autocorrelation graph in fig. 7 also shows that the values fall within the acceptable confidence limit. So, in as much as the regression plots for the training, validation and test datasets may indicate the kind of fit the data would achieve, the acceptability of the same is reinforced using the autocorrelation plot in terms of the input variables and the targets. Subsequently, the raw data was presented and analyzed in WEKA and three alternative techniques were compared. The results are presented in the figure 1.4 below.

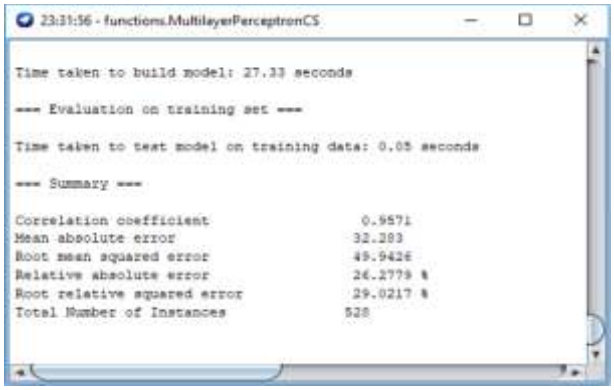

FIGURE 8 (a) Multilayer Perceptron (MLP)

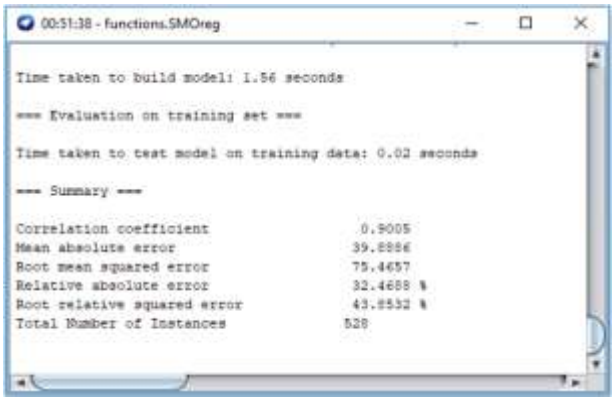

FIGURE 8 (b)Support Vector Machine (SVM)

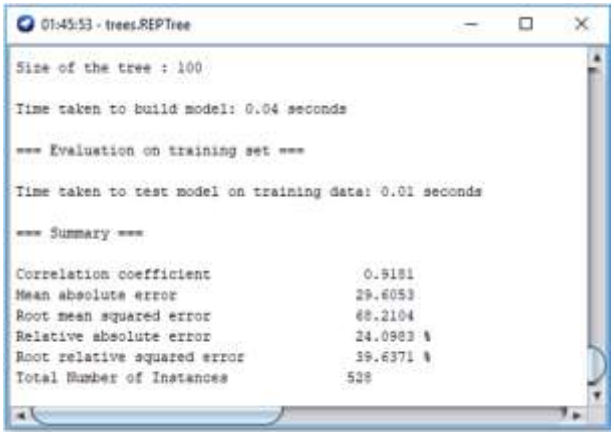

FIGURE 8 (c) Decision Tree

In fig. 8 (a), (b) and (c), represent the results obtained on execution of the MLP, SVM and decision tree techniques in WEKA. The metrics used are the Correlation coefficient, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Relative Absolute Error (RAE) and the Root Relative Squared Error (RRSE). Based on these, it is observed that these techniques have higher margins given that low values are preferable so as to invoke some confidence in their use. Also, the MLP takes a little longer compared to the other two in terms of execution time. When all these metrics are put into consideration the ideal Neural Network of choice in this case is a Neural Network with a deep architecture such as the Recurrent Neural Network.

The following table 4.0 is a tabulation of the various experiments performed during the training of the Deep Neural Network. These experiments were performed under different conditions as represented by the various fields in the table.

Five algorithms are used to train the Network and the optimal MSE are indicated in bold. These algorithms were tested when the Network was run with different sizes as indicated by the hidden layer size. The neurons in these layers are the ones that receive the source input vectors used to compute the values that are used in subsequent hidden layers and finally produce an accurate or optimal outcome based on the goal set by the cost function. On average the Scaled Conjugate Gradient Backpropagation algorithm seems to perform better than the rest. This comes at a cost of time since it takes a relatively longer than the other algorithms to finalize its computations. The other factor is that the backpropagation algorithms outperform the gradient descent algorithm which grows relatively slower over time.

TABLE 1
THE MSE OBTAINED ON TRAINING THE DEEP NEURAL NETWORK

Training Algorithm	**Hidden Layer Size**	**Performance (MSE)**				
		Training	**Validation**	**Testing**	**ClosedLoop**	**Multistep**
Levenberg-Marquardt Backpropagation	20	818.0640	783.5319	3.7364e+03	2.8123e+04	**977.3571**
	40	578.9504	**527.9206**	3.0480e+03	3.5632e+05	1.7058e+03
	60	441.7975	1.2660e+03	5.9905e+04	8.5707e+05	1.0731e+04
	80	**360.2578**	1.5782e+03	3.1874e+03	**1.1339e+06**	3.6252e+05
	100	423.0597	639.1764	**1.3035e+03**	1.3107e+05	460.7955
Bayesian Regularization Backpropagation	20	946.7266		1.3239e+03	1.4351e+03	1.1729e+03
	40	594.1056		2.0683e+03	3.7911e+05	1.3656e+03
	60	509.8641		1.5805e+03	1.6269e+05	3.6718e+04
	80	444.1823		2.4020e+03	**1.2775e+06**	1.8707e+03
	100	**438.1950**		**987.3896**	7.5681e+06	**1.0133e+05**
Scaled Conjugate Gradient Backpropagation	20	1.1593e+03	1.6378e+03	806.3088	4.1374e+04	1.0395e+03
	40	803.0078	1.4039e+03	**556.6178**	**1.1128e+04**	372.5560
	60	796.0570	**480.9656**	1.9045e+03	6.8637e+04	775.3550
	80	726.0811	580.5483	1.9324e+03	2.9323e+04	**360.8425**
	100	**634.2585**	1.8469e+03	1.3275e+03	4.9421e+04	8.1339e+03
Variable Learning Rate Gradient Descent	20	1.0230e+04	1.2907e+04	9.5923e+03	1.9384e+04	3.0298e+04
	40	1.4471e+04	6.0697e+03	2.0610e+04	2.1330e+05	2.5599e+04
	60	1.4722e+03	2.5972e+03	**1.0382e+03**	**1.4048e+05**	**801.1083**
	80	**1.2761e+03**	2.3050e+03	2.2663e+03	2.9426e+05	1.6656e+04
	100	1.6129e+03	**1.1346e+04**	2.0768e+03	4.3850e+04	9.6667e+03
Resilient Backpropagation	20	1.2708e+03	728.3699	**467.9015**	**1.6780e+04**	2.0868e+03
	40	939.4876	945.1935	1.7334e+03	1.8125e+05	1.0965e+03
	60	833.4259	1.7520e+03	546.1410	9.0586e+04	**786.1239**
	80	790.4356	**554.5862**	1.1719e+03	9.2151e+04	3.2253e+03
	100	**522.8415**	2.2746e+03	2.4530e+03	5.1805e+04	1.0100e+03

VI. CONCLUSION

Generally, from the results obtained it is possible to derive that algorithms that involve backpropagation are important in training the Deep Neural Networks. This may also come at a cost of time which can be overcome by increasing the processing power readily available in the computing world in form of parallel computing or distributed computing. Also, a gradual increase of the adjustable parameters may yield steady and reliable results especially when training the algorithm for datasets such as in the case of vehicle traffic flow. The higher the number of neurons in the hidden layers yields less errors in the context of training the Deep Neural Network. This increase is done gradually to facilitate the relative dependence of the validation and testing data set errors that are used to prevent overfitting of data. Nonetheless, the performance of the Deep Neural Networks relies on the training algorithm and the division of the dataset in order to achieve optimal performance.

REFERENCES

[1] Alfred Csikos, Zsolt Janos Viharos, Krisztian Balazs, Tamas Tettamanti and Istvan Varga (2015). Traffic Speed Prediction Method for Urban Networks – an ANN Approach, Hungary.

[2] Avigdor Gal, Avishai Mandelbaum, Francois Schnitzler, Arik Senderovich and Matthias Weidlich (2015). On Predicting Traveling Times in Scheduled Transportation, Proceedings of the 2nd International Workshop on Mining Urban Data, Lille, France, 2015.

[3] Bianchini, Monica and Scarselli, Franco (2014). On the complexity of shallow and deep neural network classifiers. ESANN 2014 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 23-25 April 2014, i6doc.com publ., ISBN 978-287419095-7.

[4] Dong Yu and Li Deng (2015). Automatic Speech Recognition: A Deep Learning Approach. ISBN 978-1-4471-5778-6. DOI 10.1007/978-1-4471-5779-3

[5] Douglas C. Montgomery, Cheryl L. Jennings and Murat Kulahci (2015). Introduction to Time Series Analysis and Forecasting, 2nd Edition. John Wiley & Sons, Inc.. ISBN-13: 978-1118745113

[6] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-Rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, and Brian Kingsbury (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition.

[7] Giuseppina De Luca and Mariano Gallo (2017). Artificial Neural Networks for forecasting user flows in transportation networks: literature review, limits, potentialities and open challenges. IEEE 978-1-5090-6484-7/17

[8] Huisken Giovanni (2006). Inter-Urban Short-Term Traffic Congestion Prediction, TRAIL Thesis Series T2006/8. The Netherlands TRAIL Research School.

[9] Jamie Houghton, John Reiners, and Colin Lim (2009). Intelligent Transport System: How cities can improve mobility, IBM Institute for Business Value, USA, 2009.

[10] Lakakis, K. and Kyriakou, K. (2015). Creating an Intelligent Transportation System for Smart Cities: Performance Evaluation of Spatial – Temporal Algorithms for Traffic Prediction, Proceedings of the 14th International Conference on Environmental Science and Technology Rhodes, Greece, 3-5 September 2015.

[11] Li Deng and Dong Yu (2013). Deep Learning: Methods and Applications. Foundations and Trends in Signal Processing Vol. 7, Nos. 3–4 (2013) 197–387. DOI:10.1561/2000000039

[12] Mahamad Nabab Alam (2016). Codes in MATLAB for Training Artificial Neural Network using Particle Swarm Optimization. DOI: 10.13140/RG.2.1.2579.3524

[13] Mark Hudson Beale, Martin T. Hagan and Howard B. Demuth (2015). Neural Network Toolbox™ User's Guide.

[14] Matthias Kormaksson, Luciano Barbosa, Marcos R. Vieira and Bianca Zadrozny (2014). Bus Travel Time Predictions Using Additive Models. IBM Research - Brazil, arXiv:1411.7973v1 [cs. LG] 28 Nov 2014.

[15] Mladen Dalto (2014), Deep Neural Networks for time series prediction with applications in ultra-short-term wind forecasting, Rn (01), Vol. 1, p. 2 (2014).

[16] Mohamed Zaki, Alaa Hamouda, and Basma Hisham (2015). Travel Time Prediction under Egypt Heterogeneous Traffic Conditions using Neural Network and Data Fusion, Egyptian Computer Science Journal Vol. 39 No. 2 May 2015 ISSN-1110-2586.

[17] Neal Wagner, Zbigniew Michalewicz, Moutaz Khouja, and Rob Roy McGregor (2005). Time Series Forecasting for Dynamic Environments: The DyFor Genetic Program Model. IEEE Transactions on Evolutionary Computation, January 2005.

[18] Nikolaus Kriegeskorte (2015). Deep Neural Networks: A New Framework for Modeling Biological Vision and Brain Information Processing. Annual Review of Vision Science 2015. 1:417–46. 10.1146/annurev-vision-082114-035447

[19] Rajkumari Bidyalakshmi Devi, Esha Barlaskar, Oinam Binarani Devi, Smriti Priya Medhi and Reingayung Ronra Shimray (2014). Survey on Evolutionary Computation Techniques and Its Application in Different Fields. International Journal on Information Theory (IJIT), Vol.3, No.3, July 2014 DOI: 10.5121/ijit.2014.3308

[20] Rathinakumar, M., Sankara Subramanian, B., Vasanth Kumar, M.R. and Kumaran, N. (2013). Auto-Mobile Vehicle Direction in Road Traffic Using Artificial Neural Networks, International Journal of Artificial Intelligence & Applications (IJAIA), Vol. 4, No. 4, July 2013.

[21] Robert Stackowiak, Venu Mantha and Art Licht (2015). Improving Logistics & Transportation Performance with Big Data. Oracle Enterprise Architecture White Paper February 2015.

[22] Seyed Mojtaba Tafaghod Sadat, Toni Anwar and Mina Basirat (2012). A Survey on Application of Artificial Intelligence for Bus Arrival Time Prediction, Journal of Theoretical and Applied Information Technology 15th December 2012. Vol. 46 No.1, ISSN: 1992-8645, E-ISSN: 1817-3195.

[23] Suyun An, Huynho Chang and Young-Ihn Lee (2013). Dynamic Forecasting of Bus Path Travel Time: Simplified Stochastic Approach, Proceedings of the Eastern Asia Society for Transportation Studies, Vol.9, 2013.

[24] Swasti Singhal, Monika Jena (2013). A Study on WEKA Tool for Data Preprocessing, Classification and Clustering. International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-6, May 2013.

[25] Takashi Kuremoto, Shinsuke Kimura, Kunikazu Kobayashi and Masanao Obayashi (2014). Time Series Forecasting Using a Deep Belief Network with Restricted Boltzmann Machines. Neurocomputing, Volume 137, 5 August 2014, Pages 47–56

[26] Tantawy, M. and Zorkany, M. (2014). A Suitable Approach for Evaluating Bus Arrival Time Prediction Techniques in Egypt. Proceedings of the 2014 International Conference on Communications, Signal Processing and Computers.

[27] Tomas Mikluscak, Michal Gregor, and Ales Janota (2012). Using Neural Networks for Route and Destination Prediction in Intelligent Transport Systems, J. Mikulski (Ed.): TST 2012, CCIS 329, pp. 380–387, 2012.

[28] Tongyu Zhu, Fajin Ma, Tao Ma, and Congcong Li (2011). The Prediction of Bus Arrival Time Using Global Positioning System Data and Dynamic Traffic Information, Wireless and Mobile Networking Conference (WMNC), 2011 4th Joint IFIP, Beijing, China.

[29] Wei-Hsun Lee, Shian-Shyong Tseng and Sheng-Han Tsai (2009). A knowledge based real-time travel time prediction system for urban network, Expert Systems with Applications 36, vol. 36, no. 3, pp. 4239–4247, 2009.

[30] Yann LeCun and Ranzato Marc'Aurelio (2013). Deep learning tutorial. In: Tutorials in International Conference on Machine Learning (ICML'13), 2013. Citeseer.

[31] Zegeye Kebede Gurmu and Wei David Fan (2014). Artificial Neural Network Travel Time Prediction Model for Buses Using Only GPS Data, Journal of Public Transportation, Vol. 17, No. 2, 2014.