SETSCI conference proceedings
SETSCI Conference Proceedings
4 (6), 384-390, 2019

4th International Symposium on Innovative Approaches in Engineering and Natural Sciences

November 22-24, 2019, Samsun, Turkey

https://doi.org/10.36287/setsci.4.6.098

ISAS WINTER - 2019

Analysis of Highway Traffic Using Deep Learning Techniques

Muhammet Esat Özdağ [1*+], Nesrin Aydın Atasoy [2]

[1] Department of Computer Technologies, Tokat Gaziosmanpaşa University, Tokat, Turkey
[2]Computer Engineering Department, Karabük University, Karabük, Turkey
[*]Corresponding author: muhammetesat.ozdag@gop.edu.tr
[+]Speaker: muhammetesat.ozdag @gop.edu.tr
Presentation/Paper Type: Oral / Full Paper

Abstract **–** Traffic flow forecasting has an important place in designing a successful intelligent transportation system. The success of the forecasting is related to the accuracy and timely acquisition of the traffic flow data. The inadequacy in the number of data has led to the use of shallow architectures in the traffic forecasting models realized so far or to design models with generated data. These models failed to occur forecast results with sufficient success. Nowadays, in the age of big data, in parallel with the increase in traffic density, there has been a significant increase in the diversity and size of the collected traffic flow data. This result constitutes the main motivation in our study. Our study aims to forecast traffic density at the exit of a motorway that have linked roads. The forecasting models proposed in our study were designed using generally accepted, Deep Learning techniques, which can occur meaningful prediction results with big data.

The techniques used in our study are Recurrent Neural Network (RNN), Long-Short Time Memory (LSTM), Stacked Long Short-Term Memory (S-LSTM), Bidirectional Long Short-Term Memory (B-LSTM) and Gated Recurrent Unit (GRU) neural networks. The dataset used in the study consists of 929 thousand 640 measurement data collected by loop sensors placed at 6 different points. Three different training data sets were created, split 90%, 80% and 70% of all data and the remainder of the data used as the test dataset. Forecast achievements of the designed models on the test dataset were recorded by calculating the Mean Square Error (MSE) values. In addition, all models are run with different number of epochs and the effect of the training set size and iterations on learning was investigated. The results show that Deep Learning techniques in traffic flow forecasting with low MSE values occur successful results and can be used in traffic flow prediction models. When the results of selected Deep Learning techniques and designed models are compared, it is observed that B-LSTM has the best forecast performance with the lowest MSE value of 36,60.

Keywords – Deep Learning, RNN, GRU, LSTM, intelligent transportation system and traffic flow forecasting

I. INTRODUCTION

Nowadays, in parallel with the increasing population, the increase in the number of vehicles increases the time people spend in traffic. In order to solve the problems caused by these increases, Intelligent Transportation Systems (ITS) start to gain more place in our lives. With the increasing use of loop sensors in the collection of traffic flow data, large amounts of real-time data are being collected. This development enables the design of ITS applications that enable people to choose better routes in traffic, or to enable authorities to make instant improvements in traffic signaling.

One of the important elements of designing a successful ITS is to know the traffic density in advance. Forecast success depends on accurate and timely data. As a result of the developments in the storage and acquisition of data and the innovations in the calculation devices, the studies in the field of estimating the traffic flow by processing real-time data are increasing day by day. The causes of traffic density do not consist of linear elements. This is the main reason why the short-term forecasting success with traditional methods is low. As a result of the increase in the volume of real-time data with temporal and spatial characteristics, deep learning, which is a pioneering technique in the interpretation of stochastic data, is beginning to rise to a frequently used position in traffic flow studies. Recurrent Neural Networks (RNN), one of the Deep Learning models, is a powerful predictive algorithm over time series. Based on this idea, in our study, we use Long short-term memory (LSTM), a type of RNNs, and its variations, and perform comparisons between variation results.

Knowing the traffic flow in advance has an important role in minimizing or eliminating traffic congestion. Since the 1990s, researchers have conducted a number of techniques ranging from Autoregressive Integrated Moving Average (ARIMA) to Artificial Neural Network (ANN) to predict traffic flow. Voort et al. proposed a model with ARIMA and Kohonen Self-Organizing Maps to estimate short-term traffic density [1]. Using only historical data from the highway in traffic flow estimation is to ignore the representations of the connection roads, which is one of the reasons for the density. Williams [2] and Shiliang et al. [3] conducted separate studies with ARIMA and Bayesian Networks, concluding that the linked roads were improved to prediction success. Choosing the right variables is important for the designed

model [4]. Kamarianakis et al. examined the effects of univariate and multivariate input data on traffic flow prediction accuracy and proved the superiority of multivariate models. [5]. Vlahogianni et al. demonstrated that it is possible to represent complex traffic data in a temporal and spatial relationship with ANN models [6] [7] [8]. Other studies have examined the effects of ANN models on predictive success by modifying parameters that represent this complex structure of traffic data [9] [10] [11]. Most of the studies for traffic flow prediction are based on eventuality and the assumption of normal ongoing traffic flow. Manoel et al. propose a model designed with Support Vector Machine (SVM), a statistical model representing weather conditions, accidents and holidays [12]. Lippi et al. developed a SVM model combined using the Kalman Filter, and found a positive correlation between seasonal effect [13] and traffic flow prediction accuracy [14]. Nowadays, in the era of big data, Yisheng et al. Based on the magnitude of training data that directly affect the success of Deep Neural Networks (DNN), he proposed a prediction model using the DNN technique Autoencoder [15].

The aim of this study is to estimate the traffic density at the motorway exit point using data collected by loop sensors placed on the UK M57 motorway and five different roads linked to this road. As a result of the literature reviews, no studies were found in traffic flow prediction studies comparing LSTM variations and the effect of hyper parameters on the results. In this respect, our study provides a new contribution to the literature.

The rest of the article continues as follows. In the next section, the techniques in the estimation model are explained. Section 3 describes the environment, the used dataset and the evaluation criteria. Section 4 contains experimental results and comparisons. The last section contains conclusions.

II. METHODOLOGY

This section discusses the used techniques and Deep Learning. II.B contains an overview of RNNs. II.C and subsequent sections provide information about LSTM and its variations.

A. Deep Learning

Deep Learning is a specific subfield of machine learning: it is defined as a new perspective that seeks to obtain a learning representation from data, focusing on more iterative and progressive learning of more consecutive strata [16].

Deep Neural Networks (DNN) contains multiple nonlinear hidden layers, and with this architecture, the network acquires a skill that can learn very complex relationships between inputs and outputs [17]. These layers are structured with layers called ANN stacked on top of each other [18]. The term Neural Network (NN) refers to neurobiology, but Deep Learning models are not a one-to-one brain model, although some of the central concepts in Deep Learning have been inspired by the human brain [18].

Automatic feature extraction is one of the great advantages that Deep Learning has over traditional machine learning algorithms [19]. Feature extraction is the process of deciding which of the properties of a data set the network can be used as an indicator for reliably labelling this data [19].

B. Recurrent Neural Network (RNN)

RNN was first introduced in the 1980s, and has gained popularity again due to recent hardware developments [20]. RNNs distinguish themselves from the Feed Forward Neural Network (FFNN) provided that it has at least one feedback loop [21]. In a repetitive NN model shown in Fig. 1, it can be seen that a neuron may have multiple connections, while at the same time the neuron may be associated with other neurons of the same layer.

Assuming that a completely recurrent NN consists of r neurons, it is assumed to have as many connections as r^2 [20].

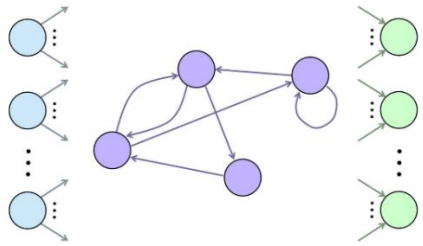

Fig. 1 A recurrent layer contains connections between neurons in the same layer [20]

RNN is a NN model developed to take advantage of sequential information and learn the patterns that exist in this information [22]. This network architecture is expected to perform the same task for all elements of an array. RNNs have been very useful in language processing problems due to the sequence dependence of words in any spoken language [22].

C. Long-Short Time Memory (LSTM)

Sepp Hochreiter and Jürgen Schmidhuber have discovered long-term network architecture as a solution to the problem of missing gradients [23]. The basic principle behind this network architecture is that the network can reliably transmit important information to the future in multiple iterations [20].

As shown in Fig. 2, the LSTM unit consists of several main components. One of the main components of the LSTM architecture is the memory cell, also known as tensor, represented by the thick frame in the middle of the figure. During many iterations, this memory cell of the network designed to store useful information stores critical information learned over time [20].

At each step, the LSTM unit changes the tensor cell in three different stages with new information. First, the unit determines how much of the previous memory it will hold. This is defined as *keep gate*.

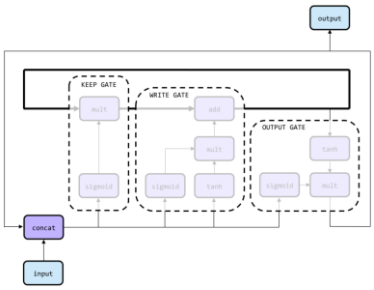

Fig. 2 Architecture of the LSTM unit shown at the tensor and process [20]

The basic idea of Keep gate is simple. The memory state tensor in the previous time step is information-rich, but some of this information may be obsolete and may need to be deleted. If the position in the bit sensor is one, it means that the position in the memory is still valid and must be stored. If the position is zero, the position in the tensor cell is no longer relevant and must be changed. The bit tensor draws that this time step input, and the output of the previous time step of the LSTM unit and bring the resulting tensor into a sigmoid layer. A sigmoidal neuron produces a value that is very close to zero or very close to one. According to this rule, the output of the sigmoidal layer is close to the working approach of the bit tensor. Therefore, this function is used to complete keep gate [20].

The information writing area of the LSTM unit is known as *write gate*. It consists of two main components. The first of these components is to find out which information is written to the tensor. This is calculated by the tanh layer. The second component is to find out which information will be discarded without writing. We make this choice with the same strategy we use in keep gate. The bit vector is multiplied by the intermediate tensor and a new state vector is created for the LSTM.

Output Gate architecture shows at Fig. 2. It is a similar structure to as write gate. The tanh layer forms an intermediate tensor from the state vector. The sigmoid layer creates a bit tensor mask using the current input and the previous output. This intermediate tensor is multiplied by the bit tensor to obtain the final output.

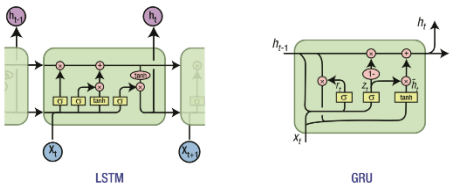

Fig. 3 LSTM and GRU structures [24]

The network created by increasing the number of hidden layers of LSTMs is defined as Stacked Long Short-Term Memory (S-LSTM). B-LSTM is a type of LSTM that previous and two series previous node connect to an output node in a hidden layer.

D. Gated Recurrent Unit (GRU)

There are many LSTM types used today. The most commonly used of these is the Gate Recurrent Unit, GRU [24]. The GRU combines forgetting and entry doors to form an update port and changes the way the output is produced. It has less complexity than standard LSTM models [25]. It has been observed that LSTM works better for larger data sets and GRU works better for smaller data sets [24].

III. Case Study

E. Data

Traffic flow data consisted of 929640 measured values collected every 15 minutes between 1 April 2015 and 31 August 2019. It was measured by several intersections (J1, J2, J3, J4, J5, J6, J7) linked to the M57 motorway in the UK and by loop sensors located at the motorway exit. The data was obtained from the publicly accessible website *tris.highwaysengland.co.uk/detail/trafficflowdata*.

The used data in the training set and the missing measurement value on a time series significantly reduces the predictive performance of the model. In our study, approximately 6% of 929640 different measurement values (i.e. 56375), could not be measurement for unknown reasons. 578 of these missing data do not match the total number of vehicles passing and the total number of segments segmented by vehicle type. In this case, the total number of segmented vehicles is considered as the total number of passes. It has been determined that a large weight of the data without measurement value is for 9 months of 2015.

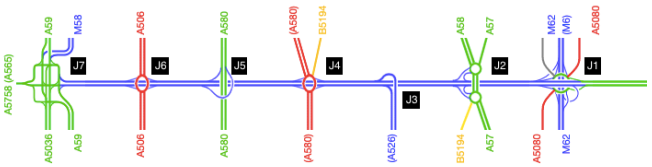

Fig. 4 M57 motorway and linked roads

In order to fill the missing data, a special program prepared in C # language was used. As the method of completion, the average values of the measurements taken from the same measurement sensors of the same day and time zone for the other 4 years were taken and the result was rounded to the small integer value and entered into the vehicle segment total field.

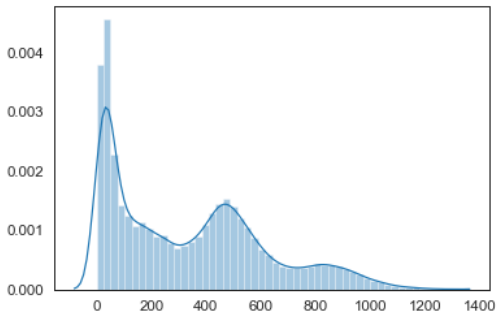

Fig. 5 Histogram distribution of vehicles exiting the M57 motorway

In Fig. 5, the p-value for D'Agostino's K-squared normality test is less than 0.05. This value indicates that the M57 motorway exit numbers do not correspond to the normal distribution, and that the motorway exit numbers may have a nonlinear dependence between the other dataset properties.

Table 1. Descriptive values

mean	Min	Max	σ	Skewness	Kurtosis	*p*
333.914	0	1283	280.079	0.636	-0.494	0.000

Normal distribution was also tested by calculating skewness and kurtosis on the data set. The skewness value being different from 0 and the kurtosis value being between -1 and +1 proves that our data does not have a Gaussian distribution. In addition, the average standard deviation to identify vehicle output data on the M57 motorway, et al. values are given in Table 1.

F. Evaluation Criteria

An NN training involves setting the link weight values [26]. The performance of the network is observed by summing the squares of the difference between the expected output value of the network and the resulting output value. This method is called MSE [26] and is defined by the formula in Equation 1:

$$f = mse = \frac{1}{N}\sum_{i=1}^{N} {e_i}^2 = \frac{1}{N}\sum_{i=1}^{N} (t_i - a_i)^2 \quad (1)$$

In this paper, the success of the designed networks is tested by calculating MSE.

G. Environment

The physical and software information of the models where the models are designed and tested is given in Table 2.

Table 2. Information about the computer on which the models are run

OS :	OS X El Captain
CPU :	2.8 GHz Intel Core i5
GPU :	Intel Iris 1516 MB
Bellek :	8 GB 1600 MHz DDR3
Keras :	Version 2.2.4
TensorFlow :	Version 1.13.1
Theano :	Version 1.0.4
Anaconda Navigator :	1.9.7

In our paper, Keras library (accessible from keras.io), which is open source and worked in TensorFlow infrastructure, was used to design and implement DNN networks. Version information is provided in Table 2.

IV. Result and Discussion

In our study, 5 different algorithms are used to estimate the traffic output density of the highway. These algorithms are RNN, LSTM, S-LSTM, B-LSTM and GRU NNs, respectively. Each algorithm was run separately by keeping the hyper-parameter different, such as the number of training iterations (also known as epoch size), the size of the training and test data set, the number of hidden layers, and the number of data simultaneously received in the training (batch size), and the results were recorded.

By using the data in the data set used in the study, 3 different data set environments were created. 90%, 80% and 70% of the training (10%, 20%, 30%) was divided into test data, with an equal amount of data in 3 environments. For models designed with different hyper parameters, these 3 sets of data sets were tried separately and the results were recorded.

In this section, the results between MSE loss function, the relationship between model run times and hyper parameters, the results of the training and the data set size, the number of iterations and the number of hidden layers are presented.

A. Results of RNN

RNN model, dataset size, epoch size, datasize entered in input layer node at once (batch size), hidden layer, number of weights (param) and node amount (node) Tested separately. The values of the generated models are as given in Table 3. Hyperbolic Tangent function was used as the activation function in all variations for this designed model. MSE was used for loss function, ADAM was used as optimization method (known as optimizer) and learning rate was kept constant as 0.001.

When the characteristics of the models are examined in Table 3, when the training dataset increases in size, the MSE value of the model decreases considerably and the predictive success increases.

When epoch size increases, the MSE values decrease, but the time spent for training increases linearly. Although there was an inverse relationship between the size of the input groups initially given to the model and time, a linear relationship with the success of the model could not be determined.

Table 3. Parameter information and results of RNN models

	Epoch Size	Batch Size	Hidden Layer	Node	Param	Size of Training Dataset	Training Time	MSE
RNN01	100	72	1	51	4001	70%	0:09:56	87.51
RNN02	100	72	1	51	4001	80%	0:15:28	69.67
RNN03	100	72	1	51	4001	90%	0:11:16	43.63
RNN04	50	72	1	51	4001	90%	0:05:36	44.47
RNN05	250	72	1	51	4001	90%	0:27:59	42.43
RNN06	500	72	1	51	4001	90%	0:52:15	42.14
RNN07	100	36	1	51	4001	90%	0:21:53	44.14
RNN08	100	144	1	51	4001	90%	0:06:04	44.16
RNN09	100	72	2	101	9051	90%	0:17:19	40.26

When the RNN model results are examined, it is seen that the model that produces the best estimation result belongs to the RNN09 architecture which has 40.26 MSE value and the number of hidden layers is increased. It has been found that increasing epoch size increases the learning, in other words, increases the accuracy of the estimation.

B. Results of LSTM

Since S-LSTM was used for multilayer LSTM, the number of hidden layers kepts constant in LSTM experiments.

Table 4. Parameter information and results of LSTM models

	Epoch Size	Batch Size	Hidden Layer	Node	Param	Size of Training Dataset	Training Time	MSE
LSTM01	100	72	1	51	15851	%70	0:21:06	98.06
LSTM02	100	72	1	51	15851	%80	0:27:44	73.41
LSTM03	100	72	1	51	15851	%90	0:24:22	40.32
LSTM04	50	72	1	51	15851	%90	0:13:07	41:38
LSTM05	250	72	1	51	15851	%90	1:02:07	39:11
LSTM06	500	72	1	51	15851	%90	1:59:59	38:66

LSTM07	100	36	1	51	15851	%90	0:45:45	40.13
LSTM08	100	144	1	51	15851	%90	0:13:50	40.59

ReLU function is used as the activation function in all variations for this designed model. MSE was used for loss function and ADAM was used as optimizer and learning rate kept constant as 0.001. When the findings were examined, it was observed that the increasing epoch size increased the educational success of the model. The LSTM06 architecture has the best MSE results.

C. Results of S-LSTM

Hyperbolic tangent function was used as activation function in S-LSTM model variations designed in our study.

Table 5. Parameter information and results of S-LSTM models

	Epoch Size	Batch Size	Hidden Layer	Node	param	Size of Training Dataset	Training Time	MSE
SK01	100	72	2	101	36051	%70	0:35:56	94.22
SK02	100	72	2	101	36051	%80	0:38:36	84.68
SK03	100	72	2	101	36051	%90	0:43:07	37.88
SK04	50	72	2	101	36051	%90	0:21:34	39.26
SK05	250	72	2	101	36051	%90	1:44:16	36.68
SK06	500	72	2	101	36051	%90	3:21:44	36.74
SK07	100	36	2	101	36051	%90	1:19:30	40.05
SK08	100	144	2	101	36051	%90	0:23:45	37.40
SK09	100	72	4	201	76451	%90	1:17:44	36.88

MSE was used for loss function, ADAM was used as optimizer and learning rate was kept constant as 0.001. When the results were examined, it was found that the best MSE value among the S-LSTM models was associated with the increase in epoch size and the number of hidden layers. S-LSTM is an LSTM model obtained by increasing the number of hidden layers in the LSTM architecture. Based on the experimental results, the architecture with the best estimation results was determined as SK05.

D. Results of B-LSTM

ReLU function was used as the activation function in all variations of B-LSTM models. MSE was used for loss function, ADAM was used as optimizer and learning rate was kept constant as 0.001.

Table 6. Parameter information and results of B-LSTM models

	Epoch Size	Batch Size	Hidden Layer	Node	Param	Size of Training Dataset	Training Time	MSE
BR01	100	72	2	101	92101	%70	1:10:01	101.48
BR02	100	72	2	101	92101	%80	1:12:30	87.18
BR03	100	72	2	101	92101	%90	1:19:17	37.46
BR04	50	72	2	101	92101	%90	0:39:09	38.46
BR05	250	72	2	101	92101	%90	3:19:04	37.67
BR06	500	72	2	101	92101	%90	6:22:14	38.85
BR07	100	36	2	101	92101	%90	2:20:29	37.98
BR08	100	144	2	101	92101	%90	0:46:54	37.71
BR09	100	72	4	201	212901	%90	2:41:46	36.60

B-LSTM is a type of LSTM that connects two previous and previous node outputs to a node in a hidden layer. When the result data were examined, it was found that the most successful estimation results among B-LSTM models were BR09 architecture obtained by increasing the number of hidden layers. Increasing batch size in the input layer increases the training performance and significantly reduces the training time.

E. Results of GRU

In our study, hyperbolic tangent function was used as activation function in all variations of GRU models. MSE was used for the loss function, ADAM was used as optimizer method and learning rate was kept constant as 0.001.

Table 7. Parameter information and results of GRU models

	Epoch Size	Batch Size	Hidden Layer	Node	Param	Size of Training Dataset	Training Time	MSE
GR01	100	72	3	151	42201	%70	0:44:05	124.68
GR02	100	72	3	151	42201	%80	0:46:44	87.01
GR03	100	72	3	151	42201	%90	0:51:13	37.58
GR04	50	72	3	151	42201	%90	0:25:27	38.53
GR05	250	72	3	151	42201	%90	2:07:02	37.62
GR06	500	72	3	151	42201	%90	4:06:14	38.95
GR07	100	36	3	151	42201	%90	1:18:56	38.08
GR08	100	144	3	151	42201	%90	0:30:13	37.75
GR09	100	72	5	251	72501	%90	1:17:48	37.71

When the results are analysed, it is seen that the model increases epoch size and the success of the prediction increases. When the MSE values were considered, it was observed that GR03 gave the best estimation result.

F. Comparison Results

In this study, 44 different models which are formed by LSTM, S-LSTM, B-LSTM, GRU and RNN techniques which are among Deep Learning techniques are designed and the success of the obtained results are compared. Table 8 gives the best results for the 5 main techniques selected model parameters are given.

Table 8. For five main RNN techniques best results of MSE

	Epoch Size	Batch Size	Hidden Layer	Node	Param	Size of Training Dataset	Training Time	MSE
GRU	100	72	3	151	42201	%90	0:51:13	37.58
B-LSTM	100	72	4	201	212901	%90	2:41:46	36.60
S-LSTM	250	72	2	101	36051	%90	1:44:16	36.68
LSTM	500	72	1	51	15851	%90	1:59:59	38:66
RNN	100	72	2	101	9051	%90	0:17:19	40.26

When the results given in Table 8 are taken into consideration, the Deep Learning Technique B-LSTM, which shows the lowest MSE value for highway traffic density estimation is the best estimation performance. In terms of training time, B-LSTM technique provided the worst performance in terms of time performance by obtaining more time than other LSTM derivatives. There was little difference between B-LSTM and S-LSTM in terms of MSE values. Despite the slight difference, S-LSTM in terms of training time has achieved a much better time performance. Even though Simple RNN MSE value is not the best value in cases where there is a restriction in training time is a technique that

can be preferred. Considering the training periods, although the MSE value is among the best, the training time of B-LSTM networks is quite high. Fig. 6, during the training iterations, the MSE values for the test and training data sets are given in a single graph.

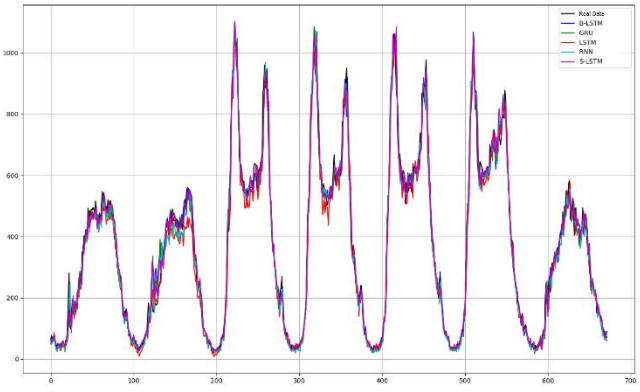

Fig. 6 Graphical representation of the actual and estimated values of the number of motorway vehicle exits in 15-minute intervals between 25 August 2019 and 31 August 2019 for the best models of the five techniques. (x axis: time, y axis: number of vehicles exiting)

When the results are analysed in terms of the techniques used in the study, they can be summarized as follows:

- Forecasting performance for models developed with RNN is a significant success with the increase in the number of hidden layers. If RNNs have the best time value in the whole technique in terms of training time, if the number of hidden layers and nodes have the same number, it will lose time because of the increase in the number of training iterations, but it does not give a significant increase in MSE values.
- In LSTM networks, it was concluded that increasing the number of training iterations increased the predictive success of the model, but there was a significant loss of training time in return. It was concluded that the number of layers used to increase LSTM success should be increased and this was observed with the model results developed with S-LSTM networks. In LSTM networks, increasing the size of the data given to the training in the input layer at a time does not increase the success of the training but provides a significant gain from the training time.
- For S-LSTM networks, it is concluded that the predictive success increases with increasing number of hidden layers and number of training iterations. Increasing the magnitude of the data given to the training at the input layer of the network at a time has a negative effect on the predictive success of the model, but the training time is greatly reduced. Although S-LSTM is the second model that has the closest value to the best model in terms of prediction performance, time has been identified as the best model for important situations.
- B-LSTM networks have been technically determined to give the best MSE value for traffic flow prediction. In order to increase the success of B-LSTM networks, it was observed that the number of hidden layers used should be increased. The most negative aspects of B-LSTM networks are the excessive size of training periods. This training time can be reduced by increasing the data size at the same time as the input layer.
- It is concluded that the number of training iterations and the size of the training data set should be increased in order to increase the predictive success of GRU models. Reducing the data size at the same time in the input layer of the network does not increase predictive success in GRU models, unlike other LSTM models.

V. Conclusions

The use of Deep Learning techniques in traffic flow prediction problems has become a frequently used method due to the fact that we are entering the age of big data and have easy access to the data. In this paper shows that the estimation of the number of vehicles to be created for multi-link motorways, so that people will be able to generate alternative road choices for the time they will spend in traffic, to ensure that the emission of emissions that cause environmental pollution that may occur due to too much traffic in the vehicles and the estimation of traffic.

Many factors, such as seasonal conditions, traffic accidents, activities and road works, affect the traffic density. Representation of these factors in the designed model will lead to an increase in forecast performance. In future studies, it will be tried to reduce the MSE value obtained by adding to the data set of traffic vehicles obtained.

References

[1] M. vanderVoort, M. Dougherty, and S. Watson, "Combining Kohonen maps with ARIMA time series models to forecast traffic flow," *Transp. Res. C, Emerging Technol.*, vol. 4, no. 5, pp. 307–318, Oct. 1996.

[2] B. M. Williams, "Multivariate vehicular traffic flow prediction—Evaluation of ARIMAX modeling," *Transp. Res. Rec.*, no. 1776, pp. 194–200, 2001.

[3] S. Sun, C. Zhang, and Y. Guoqiang, "A Bayesian network approach to traffic flow forecasting," *IEEE Intell. Transp. Syst. Mag.*, vol. 7, no. 1, pp. 124–132, Mar. 2006.

[4] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," *SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 61–72, 2004.

[5] Y. Kamarianakis and P. Prastacos, "Forecasting traffic flow conditions in an urban network—Comparison of multivariate and univariate approaches," *Transp. Res. Rec.*, no. 1857, pp. 74–84, Transporation Network Modeling 2003: Planning and Administration, 2003.

[6] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Optimized and metaoptimized neural networks for short-term traffic flow prediction: A genetic approach," *Transp. Res. C, Emerging Technol.*, vol. 13, no. 3, pp. 211–234, Jun. 2005.

[7] K. Y. Chan, T. S. Dillon, J. Singh, and E. Chang, "Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 644–654, Jun. 2012.

[8] X. Jiang and H. Adeli, "Dynamic wavelet neural network model for traffic flow forecasting," *J. Transp. Eng.*, vol. 131, no. 10, pp. 771–779, Oct. 2005.

[9] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transp. Res. C, Emerg. Technol.*, vol. 79, pp. 1–17, Jun. 2017

[10] W. Z. Zheng, D. H. Lee, and Q. X. Shi, "Short-term freeway traffic flow prediction: Bayesian combined neural network approach," *J. Transp. Eng.*, vol. 132, no. 2, pp. 114–121, Feb. 2006.

[11] B. M. Williams, P. K. Durvasula, and D. E. Brown, "Urban freeway traffic flow prediction—Application of seasonal autoregressive

integrated moving average and exponential smoothing models," *Transp. Res. Rec.*, no. 1644, pp. 132–141, 1998.

[12] M. Castro-Neto, Y.-S. Jeong, M.-K. Jeong, and L. D. Han, "Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 6164–6173, Apr. 2009.

[13] T. Maze, M. Agarwai, and G. Burchett, "Whether weather matters to traffic demand, traffic safety, and traffic operations and flow," *Transportation research record: Journal of the transportation research board*, no. 1948, pp. 170–176, 2006.

[14] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 871–882, Jun. 2013.

[15] Lv, Y., Duan, Y., Kang, W., et al.: 'Traffic flow prediction with big data: A deep learning approach', *IEEE Trans. Intell. Transp. Syst.*, 16, (2), pp. 865–873, 2015.

[16] F. Chollet and J. J. Allaire, *Deep Learning with R*, Shelter Island, NY, USA: Manning Publications Company, 2018.

[17] N. Srivastava et al., "*Dropout: A simple way to prevent neural networks from overfitting*," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[18] N. Ketkar, *Deep Learning with Python a Hands-on Introduction*, 1st ed. Apress, 2017.

[19] J. Patterson and A. Gibson, *Deep Learning: A Practitioner's Approach*, O'Reilly Media, Inc., 2017.

[20] N. Buduma and N. Locascio, *Fundamentals of Deep Learning: Designing Next-generation Machine Intelligence Algorithms*, O'Reilly Media, Inc., 2017.

[21] S. Heykin, *Neural Networks and Learning Machines (3rd Editiom)*, Pearson, New Jersey, 2009.

[22] S. Pattanayak, *Pro Deep Learning with Tensorflow: A Mathematical Aproach to Advanced Artificial Intelligence in Python*, Apress Media, pp.153-278, 2017.

[23] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] P. Goyal, S. Pandey, and K. Jain, *Deep Learning for Natural Language Processing*, Apress, 2018.

[25] J. Chung, Ç. Gülçehre, K. Cho and Y. Bengio, "*Empirical evaluation of gated recurrent neural networks on sequence modeling*," CoRR, abs/1412.3555, 2014.

[26] H. Demuth, M. Beale and M. Hagan, *User Guide, Neural Networks ToolboxTM 6*, The MathWorks, Inc., ISBN:0-9717321-0-8, 2-25, 3-5, 8-6, 2009.