Foreseeing Congestion using LSTM on Urban Traffic Flow Clusters

Ziyue Wang and Parimala Thulasiraman
Department of Computer Science
University of Manitoba
Winnipeg, Manitoba, Canada
wangz315@myumanitoba.ca, thulasir@cs.umanitoba.ca

***Abstract*—Currently, Intelligent Transportation Systems (ITS), is revolutionizing the transportation industry. ITS incorporates advanced Internet of Things (IoT) technologies to implement Smart City. These technologies produce tremendous amount of real time data from diverse sources that can be used to solve transportation problems. In this paper, we focus on one such problem, traffic congestion in urban areas. A road segment affected by traffic affects the surrounding road segments. This is obvious. However, over a period of time, other roads not necessarily close in proximity to the congested road segment may also be affected. The congestion is not stationary. It is dynamic and it spreads. We address this issue by first formulating a similarity function using ideas from network theory. Using this similarity function, we then cluster the road points affected by traffic using affinity propagation clustering, a distributed message passing algorithm. Finally, we predict the effect of traffic on this cluster using long-short term memory neural network model. We evaluate and show the feasibility of our proposed clustering and prediction algorithm during peak and non-peak hours on open source traffic data set.**

I. Introduction

Recently, intelligent transportation system (ITS) has been regarded as one of the important applications for data analytic. An ITS consists of a road network, road side units with advanced electronic sensors, intelligent control technologies in moving vehicles, wireless technologies for data transmission and so on. These technologies generate tremendous amount of data that can be collected, processed and analyzed to solve many issues in the transportation industry such as traffic flow prediction, personal travel route planning, traffic maintenance, etc.. The goal of ITS is to provide better services for personal drivers, traffic managers, emergency responders etc. to make intelligent real time decisions. We focus on the traffic flow prediction problem on a road network in urban areas.

A road network is usually represented as a graph where landmarks, junctions or intersections are used to represent the node entities in the graph and the roads/lanes are used to represent the interrelationship between the node entities. It is predicted that the traffic condition on the roads will exacerbate by 2025 with the increase in autonomous vehicles on the road. Currently, even without driver-less vehicles, traffic on the road is high. By predicting the traffic flow condition, a driver can find alternative routing paths to their destination.

In general, traffic on the road could be caused by many different reasons: accidents, construction, peak hours etc.. Assume there is an accident on the road, r, at time t. This can cause congestion on the road r and the surrounding roads near r. This is obvious and can be be predicted very easily. However, these road conditions, over a period of time, will have a domino affect, in the sense that at $t+\delta t$, the surrounding road r will affect their surrounding roads and so on. Traffic data is not only temporally correlated, it is also spatially correlated. That is, a congested road point influences the traffic on other road points.

In [27], we developed a dynamic traffic aware system using IoT technologies and sensors around road points, that dynamically collects and analyzes the traffic flow data to compute the similarity function between road points. We used the concepts from network theory, in particular maximum flow and shortest path algorithms, and a distributed, message passing technique called affinity propagation clustering, to cluster the nodes. The algorithm partitioned the road points into time variant clusters such that the traffic within the cluster are strongly, spatially correlated. The algorithm is executed continuously to capture up to date information about traffic. There are few reasons for clustering the road points. First, since the traffic road points in clusters are highly correlated, it would be easier to recognize the pattern of the traffic to further improve the solution quality. Second, clusters are disjoint. This allows easy parallelization. Third, we need to re-predict only a portion of the traffic road network reducing the problem size.

Traffic flow patterns are usually irregular and follow a time series model. In this paper, we extend the work in [27] to predict the traffic on the disjoint clusters using Long Short Term Memory (LSTM) neural network. The proposed approach will help traffic managers to monitor traffic efficiently by focusing on the areas within the clusters. The paper is organized as follows. The next section discusses the related work, followed by the background on artificial neural networks and its extensions. Section IV discusses our proposed LSTM based predictive approach. Section V describes our experiments and evaluations. Section VI concludes.

II. Related Work

Different strategies for traffic prediction have been developed and compared [15, 16, 12]. Traffic flow prediction can be divided into two categories: parametric and non-parametric techniques [15]. Traditionally, parametric time series models

978-1-7281-5255-1 ©2019 IEEE

such as Box-Jenkins, have been used to predict the traffic flow. In these models, historical traffic flow data is used in sequence of time steps to predict the future traffic flow. Autoregressive Integrated Moving Average (ARIMA) is one of the popular traffic flow prediction techniques used in parametric Box-Jenkins time series model [24, 1]. There has been many extensions of this model to improve prediction accuracy[19, 11, 21, 8, 22]. However, the results show that these classical methods have huge drawback on performance and sometimes accuracy [15, 5, 13, 23].

k-nearest neighbor (k-NN) is one of the popular non-parametric models [25, 26]. k-NN is sensitive to data and the choice of k depends on the data. Artificial Neural Network (ANN) is another popular non-parametric model, that has been considered for traffic flow prediction [2, 20]. The recent research in ANN [5, 13] have shown that variations of ANN such as convolution neural networks (CNN) and Long-Short Term Memory Neural Network (LSTM) outperform other classic parametric and non-parametric algorithms in both solution accuracy and performance.

In this paper, we propose to use Long-Short Term Memory Neural Network (LSTM) model [7], a variant of recurrent neural network. The LSTM structure is well suited for time-series data. It has been applied to traffic speed and traffic flow prediction. The authors in [14, 17] show that LSTM has better performance and solution quality over classic ARIMA model. We use LSTM on clustered data to predict the future of traffic congestion.

III. Artificial neural network

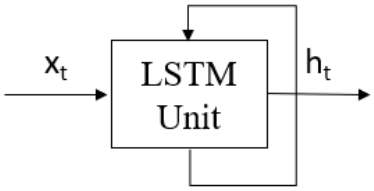
Artificial neural networks (ANN) are inspired by the neurons in the human brain. The network may have several layers consisting of neurons. Layers are directly connected to each other with weights assigned between layers. Usually, there are three types of layers: input layer, hidden layer, and output layer. Input layer is the first layer. The hidden layer does the core learning. The number of hidden layers is not restricted and depends on the user. Depending on the different type of neurons in the hidden layer, the hidden layer may have different functionality. Large number of hidden layers make the computation more compute intensive. Each layer performs a predefined activation function using the weight matrix and bias vector. Different activation functions have different functionality. For example, sigmoid function is one of the most widely used activation function. This function can take any input and convert the input into a non-linear distribution. Weight matrix assigns a weight to each input, and bias vector adds or multiplies a bias number to the final output.The final result computed by ANN and true result are compared by a loss function. Depending on the application, the loss function will result in a value that shows the accuracy of ANN. Based on this loss function, the weight matrix and bias vector are fine tuned in every iteration by the back propagation algorithm.

A. Recurrent neural network

RNN is a regular ANN where some neurons (especially in the hidden layer) have feedback connections to themselves. This connection acts as a memory element. This element takes into account the present decision, the history of decisions previously taken, to predict the future. RNN is therefore suitable for times-series data or temporal data. In our problem, the traffic data is related to both space and time.

RNN, however, fails to remember long term dependencies. To overcome this drawback, Long short-term memory (LSTM) neural network was introduced [7]. LSTM stores both long-term and short-term information. LSTM has the ability to loop back (Figure 1). It may consist of many layers. The result form h_t can loop back multiple times. The advantage of this is it can take more than one time stamp data into consideration. For example, Figure 2 shows an unrolling of LSTM neuron. x_1 is the input data at time 1. This data may affect the output result at time 3. The key idea is that the network memorizes the details to make future predictions.

Fig. 1. LSTM Unit



Among the variants of LSTM units, the Peephole [6] unit fits well in the traffic flow clustering problem. It consists of four layers: memory , input, output, and forget gate. The forget gate layer, as the name implies, forgets the unnecessary data. The Memory cell layer stores the output from the previous iteration. Input gate, forget gate, and output gate are sigmoid layers which apply sigmoid function. A sigmoid function transfers the input into sigmoid curve within the range $[-1, 1]$ (Equation 1).

$$\sigma(x) = \frac{1}{1 + e^x} \tag{1}$$

Figure 3 shows the overview of peephole LSTM structure. We use the following notations:

- Input data: $X = (x_1, x_2, \ldots, x_n)$
- Output data: $Y = (y_1, y_2, \ldots, y_n)$
- Hidden state of memory cell: $H = (h_1, h_2, \ldots, h_n)$

The current input x_t and the previous memory c_{t-1} are the inputs to forget gate. The forget gate computes the sigmoid function with these two input multiplies with a weight matrix W_f plus the bias vector b_f. The output of the forget gate f_t (Equation 2) is a value in the range $[0, 1]$, where a date value of 0 implies to remove the date and a value of 1 implies to keep the data.

$$f_t = \sigma(W_f \cdot [h_{t-1} + x_t + c_{t-1}] + b_f) \tag{2}$$

Fig. 2. Unrolled LSTM Neuron

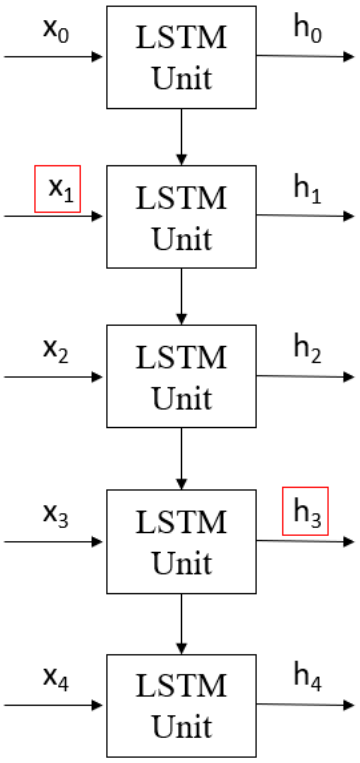
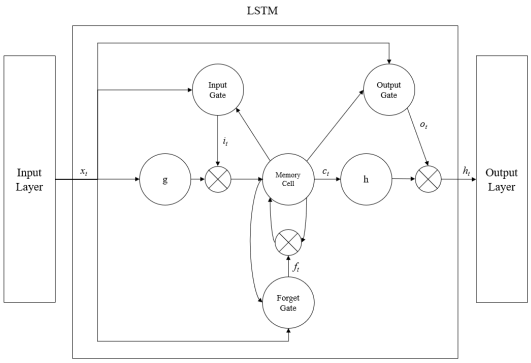


Fig. 3. LSTM Overview



The input gate (Equation 3) also takes x_t and c_{t-1} as input and decides what to store in memory for the next iteration.

$$i_t = \sigma(W_i \cdot [h_{t-1} + x_t + c_{t-1}] + b_i) \tag{3}$$

Another copy of the input is copied to the input modulation gate, which is just a modification of sigmoid function. The changes in the output, range from $[-2, 2]$. This is denoted as g. The output of the input gate and input modulation gate are added to compute c_t, the data needed to be store in memory cell (Equation 4).

$$c_t = f_t * c_{t-1} + i_t * g(W_c \cdot [h_{t-1} + x_t + c_{t-1}] + b_c) \tag{4}$$

The last step is to decide what to output. h takes the current memory c_t as input and uses the output range $[-1, 1]$. The parameter h is another modification of sigmoid function. The multiplication of O_t and $h(c_t)$ is the final output of LSTM (Equation 5).

$$h_t = o_t * h(c_t) \tag{5}$$

After we get the final output, we apply a loss function to determine the accuracy of the result. We use square loss function (Equation 6). It sums the squared error between the predicted results p_t and true result y_t.

$$e = \sum_{t=1}^{n} (y_t - p_t)^2 \tag{6}$$

We then apply back propagation algorithm to fine tune the hidden states, weights, and bias. Back propagation through time (BPTT) is a technique designed for RNN back propagation. We choose Adam optimizer [10], the combination of Adaptive Gradient algorithm and Root Mean Square propagation. It works well with sparse gradients and non-station settings. Adam optimizer requires some hyper-parameters: $\alpha, \beta_1, \beta_2, \epsilon$. We use the parameters recommended by Keras [9]: $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$. Additionally, the loss function is donated as $L(x)$ and gradient function is donated as $\nabla f(x)$. The gradient function of $f(x)$ dot product a vector v is the directional derivation of $f(x)$ along v, given in Equation 7. The Adam optimizer is given below.

$$(\nabla f(x)) \cdot v = D_v f(x) \tag{7}$$

Algorithm 1: Adam Optimizer

Data : $\alpha, \beta_1, \beta_2, \epsilon, W$
Result: W_T

$M_0 = 0$;
$R_0 = 0$;
for $t \in T$ **do**
  $M_t = \beta_1 M_{t-1} + (1 - \beta_1)\nabla L(W_{t-1})$;
  $R_t = \beta_2 R_{t-1} + (1 - \beta_2)\nabla L(W_{t-1})^2$;
  $\hat{M}_t = M_t / (1 - (\beta_1)^t)$;
  $\hat{R}_t = R_t / (1 - (\beta_2)^t)$;
  $W_t = W_{t-1} - \alpha \frac{\hat{M}_t}{\sqrt{\hat{R}_t + \epsilon}}$;
end
Return W_T;

IV. Cluster-based Traffic Prediction

In [27], we developed a dynamic traffic aware system to cluster the road points. In this section, we describe our traffic prediction on the clusters.

A. Prediction Designs

LSTM has been used in Deep learning models to extract higher level of information. Deep learning methods using

auto-encoders have been used in travel route prediction problem [12]. We use the peephole [6] unit to predict traffic. In the literature, LSTM has shown to produce good solution for time series data [4].

In general, using the time series data available for the road points, we can determine the temporal correlation between two road points. How traffic influences each road point at time t could be obtained. Since these are two different road points, we could obtain the spatial correlation between the road points. The clustered data is fed into the LSTM neural network to predict the traffic flow at time $t+\delta$. There are three prediction designs: one to one, many to one, and many to many.

Figure 4 is an example of one to one prediction: one time-series data, one prediction. For example, we use the data of the road from time t = 1,2,...,8 to predict the outcome at time 9. This is a traditional way of predicting traffic flow [5, 14, 17]. However this method does not consider the spatially correlation between road points.

Fig. 4. One to One Prediction

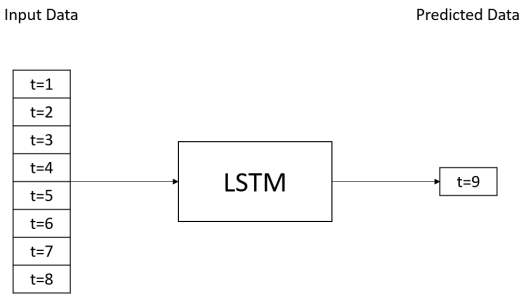


Figure 5 is many to one design: many time-series data to one prediction. Assume roads 1, 2, and 3 are in the same cluster because they have strong influence on each other. Using the data available from roads 2 and 3, it is potentially helpful to predict the outcome of road 1. In the figure, the data points for time t = 1,2,...,8 for the three roads are used to predict what happens at time 9 for road 1. This may improve the prediction accuracy but it also increases the computation overhead since more data is required to predict.

Fig. 5. Many to One Prediction

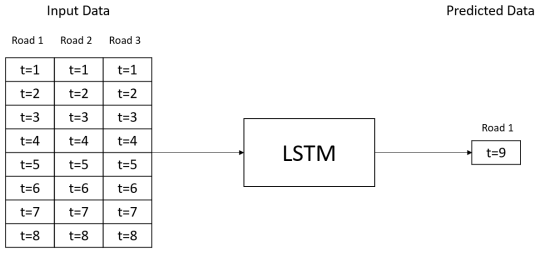
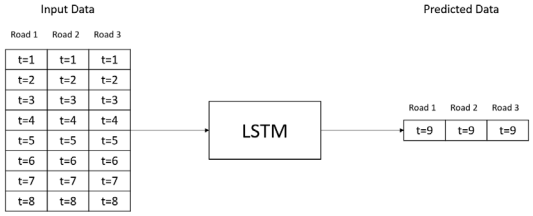


Figure 6 is many to many design. We use LSTM capabilities to predict multiple roads at a time. By feeding the data of one cluster (which includes multiple road points), LSTM can predict the outcome of the entire cluster. Compared to many to one prediction, the computation overhead is higher. However, if the prediction of the entire cluster is required, it will be more efficient since it can do the predictions in one shot rather than many to one prediction, that predicts one road at a time. On the other hand, predicting the entire cluster increases training time. We believe that with a high performance parallel computer, the training time could be reduced. This is the focus of another paper.

Fig. 6. Many to Many Prediction



B. Cluster Merge

In order to make the best use of many to one and many to many prediction model, the input time-series must have strong influences to each other.

We propose to use a cluster merge technique to merge time variant clusters. To do this, we build a similarity matrix $s[N,N]$ set to 0. Then we start to cluster the traffic from first time stamp to last time stamp in a month. If in a time stamp, a node A is in same cluster as node B, we increase the similarity of the nodes by 1, that is, $s[A,B]+=1$. After all time stamps are evaluated, we normalize the similarity matrix $s[N,N]$ by $s[N,N] = \frac{s[N,N]-min(s[N,N])}{max(s[N,N])}$ so all values in similarity matrix will be in range $s[N,N]$. The pseudocode of the algorithm is shown below.

Algorithm 2: Cluster Merge

Data : Clustering Algorithm $A(x)$, Time stamp similarities $TS = \{ts_0, ts_1, \dots ts_T\}$
Result: Cluster labels: $C(i), i \in N$

$s[N,N] = 0$;
for $t \in T$ **do**
  $C_t = A(ts_t)$;
  $\forall i,j \in N$: **if** $C_t(i) == C_t(j)$ **then**
    $s[N,N]+=1$;
  end
end
$s[N,N] = \frac{s[N,N]-min(s[N,N])}{max(s[N,N])}$;
$C = A(s)$;
Return C;

V. Experiment

The program in implemented in Python using Keras library [3] on a4.2 GHz Intel i7-7700k processor with 16 GB memory and operating system Unbuntu 18.04 LTS. The LSTM design has four hidden layers. The first two are LSTM layers with 64

LSTM units. The third layer is a dropout layer to avoid over fitting. There is a one to one mapping to the last layer to decide whether to dropout the unit in computation. The dropout rate is set to 0.2. The last layer is a Dense layer. Each neuron in this layer is connected to all neurons in dropout layer. We then apply a sigmoid function. The size of Dense layer is set to be same as the output size. In one to one and many to one model, it has only 1 unit, for many to many model, the number of units is the same as the cluster size. For the back propagation algorithm, Adam optimizer [10] is used. We use the recommended parameters from Keras [9] shown in table I. Data set is collected from CityPulse [18]. We use the data from 2014-03-01 to 2014-05-30. Data from 2014-03-01 to 2014-04-30 is used as training data, and data from 2014-05-01 to 2014-05-30 is used as test data. We use the past 1 hour data (12 time stamps) to predict the next 5 minutes (1 time stamp).

α	β_1	β_2	ϵ
0.001	0.9	0.999	10^{-8}

TABLE I
ADAM OPTIMIZER PARAMETERS

To evaluate the prediction quality, we use mean square error (MSE, equation 8), mean absolute error (MAE, equation 9), explained variance (EV, equation 10), and R^2 regression (R2, equation 11) to evaluate. MSE, MAE are the errors found in the result. We should expect a lower result. EV and R2 shows the goodness of the result. So, if the result is to be classified as good, the values should be high. The highest possible value of EV and R2 is 1, which means the predicted result is perfectly same as real data.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2 \tag{8}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - p_i| \tag{9}$$

$$EV = 1 - \frac{Var(y-p)}{y} \tag{10}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - p_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{11}$$

All three models (one to one, many to one, many to many) are evaluated. 600 echoes are executed in each training. We choose road 180709 which is one of the most busies road in data set as my experiment road. By applying affinity propagation and cluster merge algorithm, there are 51 more roads in the same cluster with road 180709. In all experiments, the time cost is about 66 μsec per step during training, and same as prediction, the time costs are same. We assume from this observation that by adding more features to the input data, the impact on performance is too insignificant to observe. In this experiment, 52 features has no impact on performance compared with only 1 feature.

Fig. 7. Prediction Result

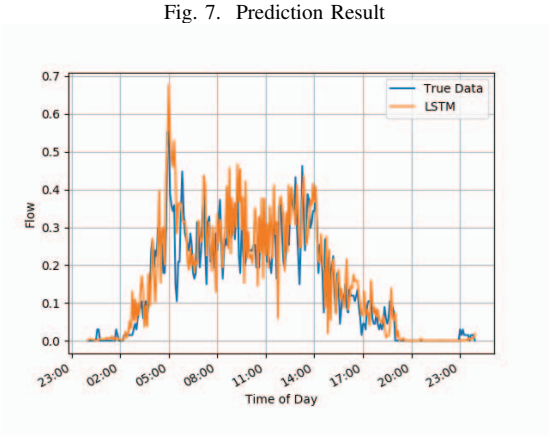


Figure 7 shows the result of many to one prediction of first day, note the traffic flow is normalized within $[0, 1]$. For peak hours, the result is not very accurate since the flow in peak hours can be extremely different depending on the day. The prediction is very close to the real traffic flow value and the floating pattern. In general, LSTM shows good solution quality.

We also evaluated all 3 models. One to one model is just normal LSTM prediction, reads road 180709 as input, and predicts road 180709 only. Many to one model reads the entire cluster of road 180709 as input, and predicts road 180709 only. Many to many model reads the entire cluster of road 180709, and predicts the future traffic for entire cluster of road 180709. The results are included in table II. In all 4 metrics, many to many achieves the best result, many to one is in second place, and one to one model is the worst. Many to one and many to many model is better than one to one model as expected. The clustered traffic data indeed improves the solution quality. MSE is reduced by 30%, MAE is reduced by 21%, the errors are clearly decreased. Both EV and R2 has increased by about 7%. Another benefit of many to many model is, it is able to predict the entire cluster, by training only once. The entire cluster data can be predicted, compared with one to one and many to one model. It saves huge amount of time in training. Since the increase in features does not increase the time cost, in this experiment, we train 52 roads together, and reduce the total time amount to $1/52$.

Models	MSE	MAE	EV	R2
One to One	0.0065	0.0517	0.7794	0.7793
Many to One	0.0045	0.0426	0.8108	0.8105
Many to Many	**0.0040**	**0.0398**	**0.8365**	**0.8335**

TABLE II
EVALUATION OF LSTM MODELS

VI. CONCLUSION

In this paper, we used the LSTM model to predict the traffic flow on the clusters. We showed the many to many design works well decreasing the error rate significantly. In the future,

we will parallelize each of the clusters on the GPU to reduce training time.

VII. Acknowledgements

The second author thanks the University of Manitoba Research Grants Program, the Faculty of Science Research Innovation and Commercialization Grant and The Natural Sciences and Engineering Research Council of Canada for their support in conducting this research.

References

[1] Mohammed S Ahmed and Allen R Cook. *Analysis of freeway traffic time-series data by using Box-Jenkins techniques*. Transportation Research Record 722. Transportation Research Board, 1979.

[2] K. Y. Chan et al. "Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm". In: *IEEE Trans. Intell. Transp. Syst.* 13.2 (June 2012), pp. 644–654.

[3] François Chollet et al. *Keras*. https://keras.io. 2015.

[4] Yanjie Duan, Yisheng Lv, and Fei-Yue Wang. "Travel Time Prediction with LSTM Neural Network". In: *IEEE 19th International Conference on ITS*. Rio de Janeiro, Brazil, Nov. 2016.

[5] Rui Fu, Zuo Zhang, and Li Li. "Using LSTM and GRU neural network methods for traffic flow prediction". In: *Chinese Association of Automation (YAC), Youth Academic Annual Conference of*. IEEE. 2016, pp. 324–328.

[6] Felix A Gers and Jürgen Schmidhuber. "Recurrent nets that time and count". In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*. Vol. 3. IEEE. 2000, pp. 189–194.

[7] S. Hochreiter and J. Schmidhuber. "Long short-term memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[8] Y. Kamarianakis and P. Prastacos. "Forecasting traffic flow conditions in an urban network—Comparison of multivariate and univariate approaches". In: *Transp. Res. Rec.* 1857 (2003), pp. 74–84.

[9] *Keras: The Python Deep Learning library*. keras. URL: https://keras.io/optimizers/#adam.

[10] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[11] S. Lee and D. Fambro. "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting". In: *Transp. Res. Rec.* 1678 (1999), pp. 179–188.

[12] Yisheng Lv et al. "Traffic flow prediction with big data: A deep learning approach." In: *IEEE Trans. Intelligent Transportation Systems* 16.2 (2015), pp. 865–873.

[13] Xiaolei Ma et al. "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction". In: *Sensors* 17.4 (2017), p. 818.

[14] Xiaolei Ma et al. "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data". In: *Transportation Research Part C: Emerging Technologies* 54 (2015), pp. 187–197.

[15] Brian L Smith, Billy M Williams, and R Keith Oswald. "Comparison of parametric and nonparametric models for traffic flow forecasting". In: *Transportation Research Part C: Emerging Technologies* 10.4 (2002), pp. 303–321.

[16] Shiliang Sun, Changshui Zhang, and Guoqiang Yu. "A Bayesian network approach to traffic flow forecasting". In: *IEEE Transactions on intelligent transportation systems* 7.1 (2006), pp. 124–132.

[17] Yongxue Tian and Li Pan. "Predicting short-term traffic flow by long short-term memory recurrent neural network". In: *IEEE International Conference on Smart City/SocialCom/SustainCom*. IEEE. 2015, pp. 153–158.

[18] Ralf Tönjes et al. "Real time iot stream processing and large-scale data analytics for smart city applications". In: *poster session, European Conference on Networks and Communications*. sn. 2014.

[19] M. vanderVoort, M. Dougherty, and S. Watson. "Combining Kohonen maps with ARIMA time series models to forecast traffic flow". In: *Transportation Research . C, Emerging Technology* 4.5 (Oct. 1996), pp. 307–318.

[20] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias. "Optimized and meta optimized neural networks for short-term traffic flow prediction: A genetic approach". In: *Transp. Res. C, Emerging Technol.* 13.3 (June 2005), pp. 211–234.

[21] B. M. Williams. "Multivariate vehicular traffic flow prediction— Evaluation of ARIMAX modeling". In: *Transp. Res. Rec.* 1776 (2001), pp. 194–200.

[22] B. M. Williams and L. A. Hoel. "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results". In: *J. Transp. Eng.* 129.6 (Nov. 2003), pp. 664–672.

[23] Billy M Williams and Lester A Hoel. "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results". In: *Journal of transportation engineering* 129.6 (2003), pp. 664–672.

[24] Billy Williams, Priya Durvasula, and Donald Brown. "Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models". In: *Transportation Research Record: Journal of the Transportation Research Board* 1644 (1998), pp. 132–141.

[25] Bin Yu et al. "k-Nearest neighbor model for multiple-time-step prediction of short-term traffic condition". In: *Journal of Transportation Engineering* 142.6 (2016), p. 04016018.

[26] Lun Zhang et al. “An improved k-nearest neighbor model for short-term traffic flow prediction”. In: *Procedia-Social and Behavioral Sciences* 96 (2013), pp. 653–662.

[27] Parimala Thulasiraman Ziyue Wang and Ruppa K. Thulasiram. “A Dynamic Traffic Awareness System for Urban Driving”. In: *IEEE Cybermatics Congress*. Atlanta, USA, July 2019.