Traffic Flow Forecasting using Multivariate Time-Series Deep Learning and Distributed Computing

Ngoc-Phap Trinh[1,2,*], Anh-Khoa N. Tran[1,2,*], and Trong-Hop Do[1,2,†]

[1]University of Information Technology, Ho Chi Minh City, Vietnam
[2]Vietnam National University, Ho Chi Minh City, Vietnam
Email: *{18521227, 18520938}@gm.uit.edu.vn
†hopdt@uit.edu.vn

***Abstract*—Traffic flow prediction is one of the most important and challenging problems. In this study, we built several univariate and multivariate time series models including LSTM, TCN, Seq2Seq, NBeats, ARIMA and Prophet using distributed deep learning to deal with the traffic flow prediction problem. The models are implemented and their performances were evaluated on a dataset of traffic flows in Ireland. The proposed multivariate models take the combination of traffic flow data, weather in the local area, and graph data of connections between traffic positions to produce the prediction of the traffic flow. The experimental results show that the proposed multivariate deep learning models achieved better prediction accuracy compared to the univariate models and machine learning models. Several other experiments were also conducted to examine the performances of these models in different scenarios to help understand more about the performance of these models.**

***Keywords*—**Multivariate time series, Traffic flow forecasting, Weather, Graph, Big Data, Deep Learning, Distributed Computing

I. Introduction

Traffic congestion has become a big problem worldwide with rapid vehicle growth and urbanization nowadays. It exacerbates pollution emissions and leads to low efficiency of the road network. Traffic flow information with accuracy and in time is an urgent need for individual travellers, the business sectors, and government agencies [1]. It plays an important role in helping road users make better travel decisions, alleviate traffic congestion, public travel safety, reduce carbon emissions and improve the efficiency of transport operations. The goal of traffic flow prediction is to provide such traffic flow information. With the rapid development and deployment of intelligent transportation systems (ITS), traffic flow prediction has received increasing attention. It is considered a key element in the successful deployment of ITS subsystems, especially traffic management systems, mass transit systems and commercial vehicle operations. Traffic flow forecasting relies heavily on historical and real-time traffic data collected from a variety of sensor sources, including inductive loops, radars, cameras, mobile GPS, crowd-sourcing, social media, and more. Traffic data are currently exploding with the appearance of new emerging traffic sensor technologies accompany by the current widespread traditional traffic sensors, and we have entered the era of Big Data transportation. Transportation management and control nowadays are becoming more data-driven [2], [3]. Research on forecast traffic flow in urban areas is thus crucial and it has been regarded as the most important issue of intelligent transport management since it can lead to scientific decisions on the guidance of effective traffic control. In the last decades, concepts of traffic bottleneck and traffic flow forecasting have been considered in many studies.

The most representative data-driven approach is the neural network and deep learning [4], [5], [6], which can automatically extract the relevant high-level features of traffic flow data. Recently, deep learning has proven to be successful in many areas such as image, audio and language learning tasks [7], [8]. For traffic congestion analysis and traffic flow forecasting, the deep learning methods have also aroused enormous research interest in recent years. The appearance of the distributed deep learning library BigDL allows for training models with a large-scale dataset to resolve the traffic flow prediction problem that cannot be done by local model systems. Based on BigDL, we build baseline models ARIMA, Prophet and deep learning models such as LSTM, Seq2Seq, Nbeats and TCN in both univariate and multivariate cases. We will train the models, compare and evaluate the performance between the baseline models and the deep learning model as well as the performance between univariate and multivariate learning based on the dataset we built through data collection, integration, and extraction process with the information about traffic, weather, the relation between sensors and its route, etc. The goal of this topic is to find the model with the highest performance, suitable to solve this problem after

experiencing the process of testing and tuning. The model input is data consisting of a time value along with traffic flow related data is the number of vehicles traveling on the route, and with associated attributes such as weather data, data about the relationship between sensors on the routes, etc in the N time steps (the multivariable model will use the associated attributes). The output to be noticed is the predictive value of data related to traffic flow is number of vehicles traveling on the route from time point N+1 onwards.

The contributions of the paper can be summarized as follows. First, we built a dataset which combines traffic data, weather, and graph data showing the relationship between traffic locations and routes. Second, based on this dataset, we proposed multivariate deep learning time series models to predict traffic flow. Third, we conducted several experiments with various deep learning and machine learning univariate and multivariable time series models and compared their performances. The proposed multivariate deep learning model that utilizes weather and graph data achieved the best performance. We also examined the performances of these models in several experimental scenarios. All models are trained using distributed learning so that the huge dataset can be utilized for training data.

II. Related work

S Du et al. (2017) [9] propose a hybrid deep learning framework for short-term traffic flow forecasting. It is built by the multilayer integration deep learning architecture and jointly learns the spatial-temporal features. The framework consists of Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). The experimental results indicate that the hybrid framework is capable of dealing with complex nonlinear urban traffic flow forecasting with satisfying accuracy and effectiveness.

Kang et al. (2017) [10] employ the long short-term memory (LSTM) recurrent neural network to analyze the effects of various input settings on the LSTM prediction performances. Flow, speed, and occupancy at the same detector station are used as inputs to predict traffic flow. The results show that the inclusion of occupancy/speed information may help to enhance the performance of the model overall.

Y Jia et al. (2017) [11] introduced the deep belief network (DBN) and long short-term memory (LSTM) to predict urban traffic flow considering the impact of weather data - rainfall. Experimental results indicate that, with the consideration of additional rainfall factors, the deep learning predictors have better accuracy than existing predictors and also yield improvements over the original deep learning models without rainfall input.

Wu et al. (2018) [12] proposed the Graph Attention LSTM Network (GAT-LSTM) by extending the LSTM to have graph attention structure in both the input-to-state and state-to-state transitions and using it to build an end-to-end trainable encoder-forecaster model to solve the multi-link traffic flow forecasting problem. Experiment results show that our GAT-LSTM network could capture spatio-temporal correlations better and improved 15% - 16% over the state-of-the-art baseline.

Zhao et al. (2019) [13] propose a deep learning framework based on the TCN model for short-term city-wide traffic forecast to accurately capture the temporal and spatial evolution of traffic flow. Moreover, the authors design the model with the Taguchi method to develop an optimized structure of the TCN model. The experimental results demonstrate that the framework achieves state-of-the-art performance.

Lu et al. (2020) [14] proposed a combined prediction method for short-term traffic flow based on the autoregressive integral moving average (ARIMA) model and long short-term memory (LSTM) neural network. The method could make short-term predictions of future traffic flow based on historical traffic data. The experimental results show that the dynamic weighted combination model proposed has a better prediction effect when compared with the three comparative baselines of ARIMA and LSTM two single methods and an equal weight combination.

III. Dataset

In this section, we present basic information about the dataset and collection process. We collect data from the TII Traffic Data website including vehicle traffic data on roads in Ireland. The TII Traffic Data website presents data collected from the TII traffic counters located on the road network. We only select data on vehicle count recorded from 11 sensors in 2021. Each record is separated by one hour and shows the number of vehicles in one hour with one direction at one sensor, each sensor records vehicle data counts in both directions.

In addition, we incorporate weather data into this traffic data to obtain additional features for multivariate prediction. The weather data is collected from the Met Éireann website. We choose the data collected from the sensor with the closest coordinates to the 11 traffic data collection sensors mentioned above for the most accurate results. Finally, we obtain a traffic dataset with 192,720 data points.

IV. The Methodologies

The methods are approached and experimented with the BigDL framework. Figure 1 shows the overview of the proposed traffic flow forecasting pipeline.

A. Data preprocessing and EDA/Feature engineering

Figure 2 shows the map with the locations of sensors. Green points represent the sensors, blue points represent intersection points of the sensors, the A and B suffixes of each sensor represent directions.

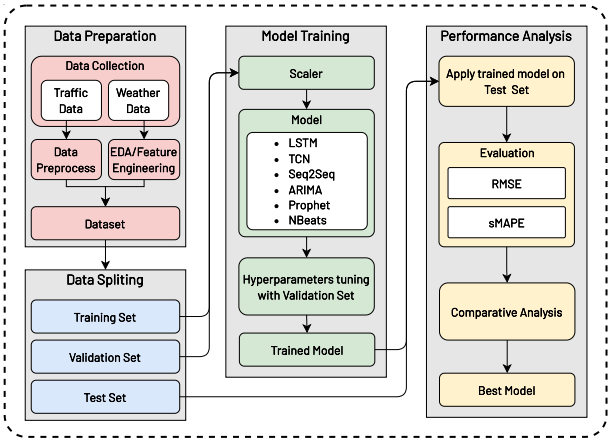

Fig. 1: The time-series forecasting pipeline.

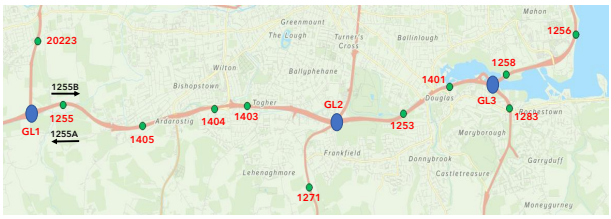

Fig. 2: The map showing the location of the sensors.

The raw traffic dataset after a few preprocessing steps consists of only the fields "datetime" (year, month, day, hour), "id" (id of sensors), "value" (vehicles count), "location" (latitude and longitude of sensors). Through observing the map, we see the connection of sensors, so we decide to integrate this feature into the dataset. A list of fields we add to represent the connectivity of the sensors on the roads, our assumption is that if there is a connection between the two sensors, the number of vehicles recorded by those two sensors influences each other. The list of fields is added as a dataframe like a matrix with the values 0 or 1 as illustrated in Figure 3 representing the link between sensors, the columns represent the sensors or the intersection points of the sensors, the indexes represent the direction of movement at the sensors.

Id	20223	GL1	1255	1405	1404	1403	GL2	1271	1253	1401	GL3	1283	1258	1256
000000020223A	1	1	0	0	0	0	0	0	0	0	0	0	0	0
000000020223B	1	0	0	0	0	0	0	0	0	0	0	0	0	0
000000001255A	0	1	1	0	0	0	0	0	0	0	0	0	0	0
000000001255B	0	0	1	1	0	0	0	0	0	0	0	0	0	0
000000001405A	0	0	0	1	1	0	0	0	0	0	0	0	0	0
000000001405B	0	0	1	1	0	0	0	0	0	0	0	0	0	0
000000001404A	0	0	0	1	1	0	0	0	0	0	0	0	0	0
000000001404B	0	0	0	0	1	1	0	0	0	0	0	0	0	0
000000001403A	0	0	0	0	0	1	1	0	0	0	0	0	0	0
000000001403B	0	0	0	0	1	1	0	0	0	0	0	0	0	0
000000001271A	0	0	0	0	0	0	1	1	0	0	0	0	0	0
000000001271B	0	0	0	0	0	0	0	1	0	0	0	0	0	0
000000001253A	0	0	0	0	0	0	0	0	1	1	0	0	0	0
000000001253B	0	0	0	0	0	0	1	0	1	0	0	0	0	0
000000001401A	0	0	0	0	0	0	0	0	1	1	0	0	0	0
000000001401B	0	0	0	0	0	0	0	0	0	1	1	0	0	0
000000001283A	0	0	0	0	0	0	0	0	0	0	1	1	0	0
000000001283B	0	0	0	0	0	0	0	0	0	0	0	1	0	0
000000001258A	0	0	0	0	0	0	0	0	0	0	1	0	1	0
000000001258B	0	0	0	0	0	0	0	0	0	0	0	0	1	1
000000001256A	0	0	0	0	0	0	0	0	0	0	0	0	1	1
000000001256B	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fig. 3: Link matrix extracted from sensors location.

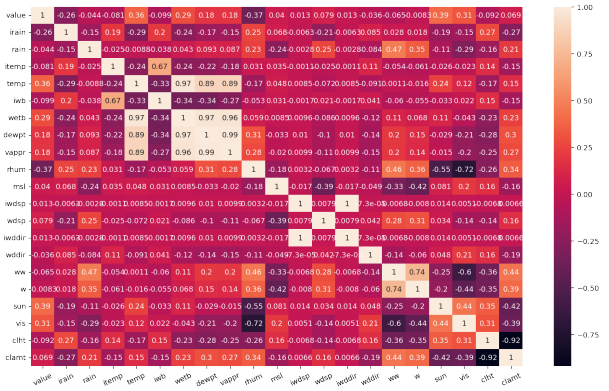

Fig. 4: Correlation matrix of the weather data.

In terms of weather data, we only extract a few fields that have the most influence on the value to be predicted. By EDA methods, we obtain 3 fields that have the best correlation with the prediction variable: "rhum" (Relative Humidity), "vis" (Visibility), "sun" (Sunshine duration) as described in Figure 4. Finally, a complete time-series dataset is added with weather, coordinate, and connectivity features.

B. BigDL

The common feature of traffic datasets is their large size, so to train a model on this type of data requires a big data processing tool, BigDL [15] is one of them. BigDL - a distributed deep learning framework for Apache Spark, which has been used by a variety of users in the industry for building deep learning applications on production big data platforms. It allows deep learning applications to run on the Apache Hadoop/Spark cluster so as to directly process the production data, and as a part of the end-to-end data analysis pipeline for deployment and management. Unlike existing deep learning frameworks, BigDL implements distributed, data parallel training directly on top of the functional compute model (with copy-onwrite and coarse-grained operations) of Spark.

C. Models

BigDL makes it easy for data scientists and data engineers to build end-to-end, distributed AI applications. BigDL provides Chronos - an application framework for building large-scale time series analysis applications. Chronos features several built-in Deep Learning and Machine Learning models for time series forecasting, detection, and simulation as well as many data processing and feature engineering utilities. In this study, we use the models in Chronos library including: ARIMA [16], Prophet [17], LSTM [18], TCN [19], Seq2Seq [20], NBeats [21] as described in Figure 5.

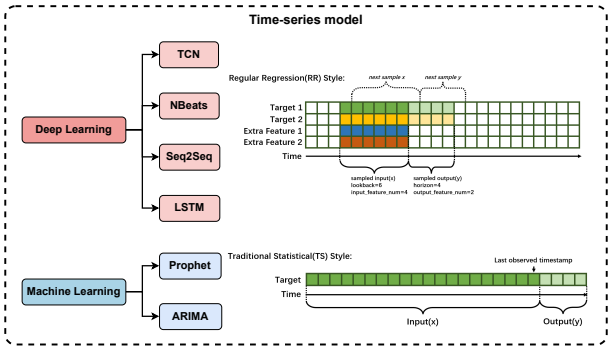

Fig. 5: Time-series model.

V. Experiments

A. Data splitting

We divide the original data set of 192,720 records into three sets: training set, validation set and test set with the ratio 8:1:1. The training set is used to train the models, the validation set is used to tune the model parameters to find the best set of parameters and the test set is used to evaluate the predictive performance of the best model. We use the Min-Max Scaler to normalise the field of traffic count value as input to train the models.

B. Evaluation metrics

The forecasting performance of the various models was evaluated using two summary statistics: root mean square error (RMSE) and symmetric mean absolute percentage error (sMAPE). Symmetric mean absolute percentage error (sMAPE) is an accuracy measure based on percentage (or relative) errors. Relative error is the absolute error divided by the magnitude of the exact value. In contrast to the mean absolute percentage error (MAPE), sMAPE has both a lower bound and an upper bound. Since it's percentage-based, it's scale-independent, which means that it can be used to compare forecast performances between datasets. It is usually defined as Eq. 1:

$$sMAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|F_t - A_t|}{(|A_t| + |F_t|)/2}, \tag{1}$$

where A_t is the actual value and F_t is the forecast value.

The absolute difference between A_t and F_t is divided by half the sum of absolute values of the actual value A_t and the forecast value F_t. The value of this calculation is summed for every fitted point t and divided again by the number of fitted points n.

Root mean square error (RMSE) is a quadratic scoring rule which measures the average magnitude of the error. The equation for the RMSE is given in both of the references. Expressing the formula in words, the difference between forecast and corresponding observed values are each squared and then averaged over the sample. Finally, the square root of the average is taken. Since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors. This means the RMSE is most useful when large errors are particularly undesirable. The RMSE value can be expressed as Eq. 2:

$$RMSE = \sqrt{\sum_{i=1}^{n}\frac{(\overline{y_i} - y_i)^2}{n}}, \tag{2}$$

where $\overline{y_i}$ is the prediction and y_i is the true value.

C. Result and evaluate

Firstly, we compare the performance of the models with the training-validation-test set ratio value of 8:1:1 to evaluate the overall results of the models. The models evaluated include LSTM, TCN, Seq2Seq, NBeats, Prophet and ARIMA. The models are trained on both univariable and multivariable. For univariable, the model's input consists of only one column "datetime", while multivariable includes variables about coordinates, weather and connectivity. The Table I below shows the number of features of the models, the value "None" represents no features added other than "datetime" (univariate model), "location_ft + link_ft" shows that in addition to "datetime", there are also features about coordinates and features about the interconnection between sensors that are applied as input to the model.

TABLE I: Predictive performance of univariate and multivariate models with RMSE and sMAPE measure.

Additional Features	Metric	LSTM	TCN	Seq2Seq	NBeats	Prophet	ARIMA
None	**RMSE**	130.24	398.54	159.58	144.92	1172.85	1292.49
	sMAPE	0.28	0.47	0.27	0.38	1.03	1.27
location_ft	**RMSE**	134.04	416.27	151.26	-	-	-
	sMAPE	0.24	0.52	0.25			
link_ft	**RMSE**	125.05	144.81	145.58	-	-	-
	sMAPE	0.28	0.31	0.24			
weather_ft	**RMSE**	131.52	143.29	153.19	-	-	-
	sMAPE	0.28	0.36	0.26			
location_ft + link_ft	**RMSE**	124.38	164.89	144.75	-	-	-
	sMAPE	0.28	0.37	0.25			
location_ft + weather_ft	**RMSE**	122.49	140.52	150.30	-	-	-
	sMAPE	0.26	0.30	0.24			
link_ft + weather_ft	**RMSE**	124.16	141.25	154.63	-	-	-
	sMAPE	0.23	0.33	0.26			
location_ft + link_ft + weather_ft	**RMSE**	**121.69**	139.00	141.63	-	-	-
	sMAPE	**0.21**	0.31	0.23			

The results in Table I show that traditional models like ARIMA, and Prophet give significantly lower results compared to other modern models because of their simple architecture. Modern models give high accuracy results, especially LSTM. Besides, the use of additional features has improved the model performance quite significantly compared to training univariate models. Therefore, the addition of features brought effective results in this study. We obtain the best model which is LSTM combined with all additional features with the result RMSE value is 121.69 and sMAPE is 0.21.

After obtaining the best model as LSTM, we continued to experiment with comparisons between the best model and the baseline model in different specific aspects.

In the first experiment, we wanted to know the difference in the performance of the model when predicting the number of vehicles at different sensor locations. The

results in Figure 6 show that the ARIMA model has a higher prediction error than the LSTM model at both metrics, the LSTM model gives more accurate and stable prediction results with low and similar prediction error values in sensors located at many different routes.

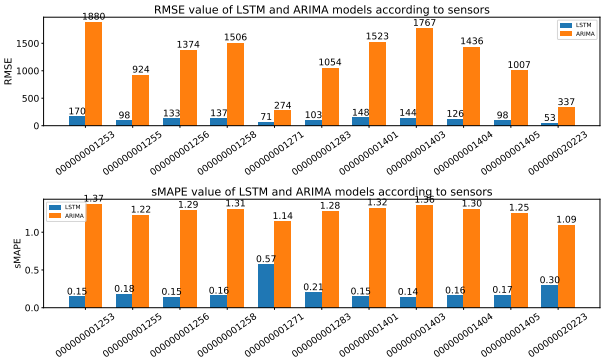

Fig. 6: Prediction error at different sensors.

In the second experiment, we compared the model performance when predicting the number of vehicles at different time points. The results obtained in Figure 7, at both metrics show that the LSTM model always gives better accurate prediction results than the ARIMA model with lower prediction error at every time point. The difference in prediction error between the two models is quite large in the period from 6 am to 8 pm (this is the peak time of the day with a large amount of traffic on the roads). During this time period, the ARIMA model has a very high increase in prediction error compared to other times of the day. The difference in prediction error between the 2 models tends to decrease after that period of time. The LSTM model still shows stability in prediction results with low prediction error and is quite similar at different times of the day.

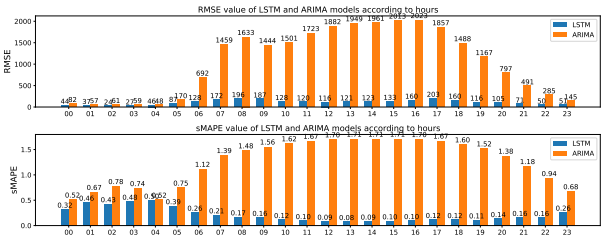

Fig. 7: Prediction error at different day hours.

Next, we compared the model performance when predicting the number of vehicles on weekdays. The results in Figure 8 at both RMSE and sMAPE metrics show that the LSTM model still has superior performance compared to the traditional ARIMA model. On different days of the week, both models show stability in prediction results with similar prediction error values in both metrics.

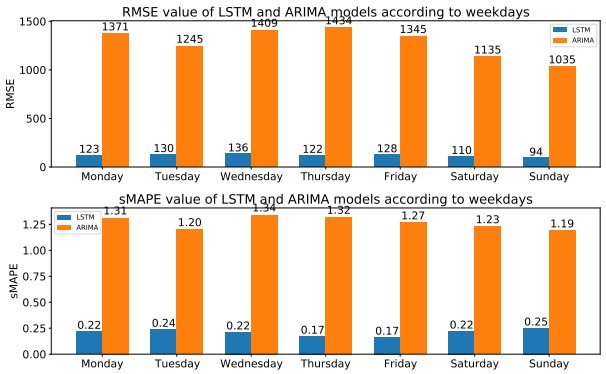

Fig. 8: Prediction error on different weekdays.

We continued to compare the forecast performance between the two models when the length of the training set is different. We train the model on four training sets of 20%, 40%, 60% and 80% of the full dataset. The results in Figure 9 show that the line histogram of the LSTM model tends to go down in the direction of increasing the data size, while the ARIMA model's one goes down in each length 20%, 40%, 60%, however, increases at 80%. In general, when training the time-series model with more data and the more timely the data, the better the prediction results. It can be seen in this experiment that the results of the LSTM model improve much when providing more training data. ARIMA model is similar, but there is an exception in the case of training data of 80% full data. This clearly shows the advantage of the deep learning model, the more data, the higher the accuracy, and ARIMA only learns the trend, the more input data will confuse the trend, leading to reduced accuracy.

In the last experiment, we compared the performance of models trained with four datasets of the same size but different in the data timeliness compared to prediction time. The results in Figure 10 show that the LSTM model still shows accuracy and stability in the prediction results compared to the ARIMA model with lower prediction error and quite similar when trained on four different datasets. The more timeliness the training dataset compared to the test dataset time, the more decrease in RMSE, sMAPE metrics of the LSTM and ARIMA models. Both LSTM and ARIMA models achieve the best prediction results in the case trained on the dataset that has the timeliness closest to the prediction time.

VI. Conclusion

In this paper, we have successfully trained a multivariate time series model using distributed deep learning to solve the traffic forecasting problem. The process of training, testing, and evaluation is performed on the actual traffic flow data set combining weather information and the graph information between traffic locations collected by ourselves. The obtained results, the multivariate model

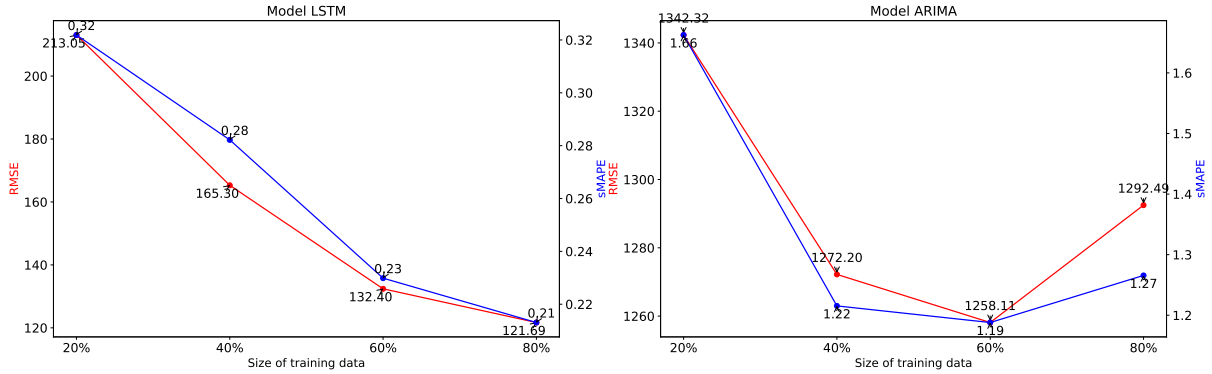

Fig. 9: Prediction error with different training data sizes.

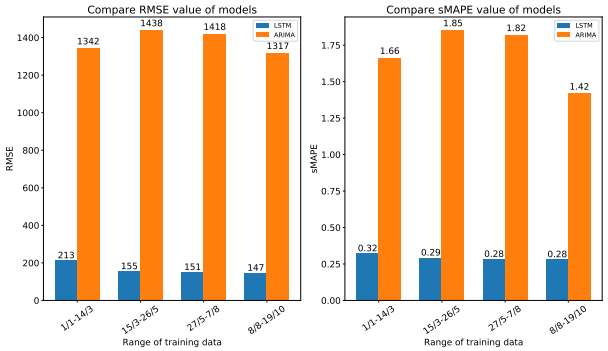

Fig. 10: Prediction error with the difference in the timeliness of training data.

with the full combination of input attributes always gives the best results in the models, proving that we have effectively exploited spatial-temporal interdependence. Based on the experimental results, we show that the traffic flow prediction results can be significantly improved with the multivariate learning model based on the dataset with the combination of features. That proves traffic conditions are related to the flow of moving vehicles, events, weather and especially the traffic flow situation between adjacent traffic network nodes are interdependent.

ACKNOWLEDGMENT

This research was funded by University of Information Technology - Vietnam National University HoChiMinh City under grant number D1-2022-48.

REFERENCES

[1] N. Zhang, F.-Y. Wang, F. Zhu, D. Zhao, and S. Tang, "Dynacas: Computational experiments and decision support for its," *IEEE Intelligent Systems*, vol. 23, 2008.

[2] C. Chen and C.-Y. Zhang, "Data-intensive applications, challenges, techniques and technologies: A survey on big data," *Information Sciences*, vol. 275, p. 314–347, 08 2014.

[3] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398, 2019.

[4] Y. Chen, L. Shu, and L. Wang, "Poster abstract: Traffic flow prediction with big data: A deep learning based time series model," pp. 1010–1011, 05 2017.

[5] H. Yi, H. Jung, and S. Bae, "Deep neural networks for traffic flow prediction," in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 328–331, 2017.

[6] J. Zheng and M. Huang, "Traffic flow forecast through time series analysis based on deep learning," *IEEE Access*, vol. PP, pp. 1–1, 04 2020.

[7] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, 04 2014.

[8] M. R. Minar and J. Naher, "Recent advances in deep learning: An overview," 02 2018.

[9] S. Du, T. Li, X. Gong, Y. Yang, and S.-J. Horng, "Traffic flow forecasting based on hybrid deep learning framework," *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pp. 1–6, 2017.

[10] D. Kang, L. Yisheng, and Y.-y. Chen, "Short-term traffic flow prediction with lstm recurrent neural network," pp. 1–6, 10 2017.

[11] Y. Jia, J. Wu, and M. Xu, "Traffic flow prediction with rainfall impact using a deep learning method," *Journal of Advanced Transportation*, vol. 2017, pp. 1–10, 08 2017.

[12] T. Wu, F. Chen, and Y. Wan, "Graph attention lstm network: A new model for traffic flow forecasting," pp. 241–245, 07 2018.

[13] W. Zhao, Y. Gao, T. Ji, X. Wan, F. Ye, and G. Bai, "Deep temporal convolutional networks for short-term traffic flow forecasting," *IEEE Access*, vol. PP, pp. 1–1, 08 2019.

[14] S. Lu, Q. Zhang, G. Chen, and D. Seng, "A combined method for short-term traffic flow prediction based on recurrent neural network," *Alexandria Engineering Journal*, vol. 60, 07 2020.

[15] Jason, Dai, Y. Wang, X. Qiu, D. Ding, Y. Zhang, Y. Wang, X. Jia, Cherry, Zhang, Y. Wan, Z. Li, J. Wang, S. Huang, Z. Wu, Y. Wang, Y. Yang, B. She, D. Shi, and G. Song, "Bigdl: A distributed deep learning framework for big data," 04 2018.

[16] J. Franklin, "A time series model for the stochastic process associated with acoustic measurement systems.," in *ICASSP '77. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 303–306, 1977.

[17] S. J. Taylor and B. Letham, "Forecasting at scale," *The American Statistician*, vol. 72, pp. 37 – 45, 2017.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[19] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *CoRR*, vol. abs/1803.01271, 2018.

[20] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *CoRR*, vol. abs/1409.3215, 2014.

[21] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: neural basis expansion analysis for interpretable time series forecasting," *CoRR*, vol. abs/1905.10437, 2019.