Comparing two hybrid neural network models to predict real-world bus travel time

Nimpanomprasert, Thummaporn; Xie, Lin; Kliewer, Natalia

Published in:
Transportation Research Procedia

DOI:
10.1016/j.trpro.2022.02.049

Publication date:
2022

Document Version
Publisher's PDF, also known as Version of record

Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 62 (2022) 393–400

Transportation Research Procedia

www.elsevier.com/locate/procedia

24th Euro Working Group on Transportation Meeting, EWGT 2021, 8-10 September 2021, Aveiro, Portugal

Comparing two hybrid neural network models to predict real-world bus travel time

Thummaporn Nimpanomprasert[a], Lin Xie[a,*], Natalia Kliewer[b]

[a]*Leuphana University of Lüneburg, Universitätallee 1, 21335 Lüneburg, Germany*
[b]*Freie Universität Berlin, Garystr. 21, 14195 Berlin, Germany*

Abstract

In order to enhance the efficiency and reliability of bus transportation systems, it is important to predict travel time precisely in the planning phase. With the precise prediction of bus travel times, the cost can be reduced for bus companies, such as from planning fixed buffer times between trips, while a better service can be provided for passengers. In this study, historical bus travel data from a bus line of the HOCHBAHN bus company in Germany in 2019 are used for the analysis. We develop two neural network models, namely multilayer perceptrons and a long short-term memory neural network, for predicting the bus travel time between timepoints for a trip occurring at a given time of day, on a given day of the week, and in given weather conditions. Both neural networks are combined with a genetic algorithm and a Kalman filter to improve accuracy. The genetic algorithm is implemented to tune the neural network parameters, such as the number of hidden layers and neurons in a hidden layer, while the Kalman filter algorithm is used to adjust the travel time prediction for the next trip using the latest bus operation information. In the experiment, we test our models month by month and split data for each month into three parts: the data of the first two weeks for training, one week for validation and the last week for prediction. The experiment results show that the hybrid model is powerful for predicting the bus travel time. In particular, the combination of the multilayer perceptrons with the genetic algorithm and Kalman filter provides the best travel time prediction (with an improvement of 56.2% compared with the planned bus travel time from the bus company). In order to make a recommendation for bus companies to plan vehicles with more accurate bus travel times, we test our hybrid models with larger training sets to predict the travel times e.g. in August. However, due to the structure of the planning problem, the plan for buses should be estimated before a new month begins. We cannot consider the real information during the planning, therefore we apply our hybrid models without the Kalman filter and they provide on average about 26% improvement compared with the planned travel time.

Keywords: bus travel time predication; multilayer perceptrons; long short-term memory neural network; genetic algorithm; Kalman filter

* Corresponding author. Tel.: +49-4131-677-2305.
E-mail address: xie@leuphana.de

10.1016/j.trpro.2022.02.049

1. Introduction

In everyday operation, the plans for buses cannot be kept as planned, e.g. due to delays, which might be caused by traffic jams or weather. Bus companies either plan some fixed buffer time to absorb the delays, or recover the plans in the operation. Both methods are expensive, since they cause further vehicle and personnel costs. In order to reduce costs for the bus company and provide a better service for passengers, it is important to predict bus travel times precisely.

Bus travel time forecasting has been widely studied using various methods by many researchers, including a historical model based on historical travel time, a dynamic model based on GPS data, a statistical estimation model (e.g. a Kalman filter), a machine learning model (e.g. long short-term memory neural network (LSTM)) and a hybrid model. An overview of the existing methods used to estimate the arrival time / travel time of buses can be found in Altinkaya and Zontul (2013), Choudhary et al. (2016) and Agrawal (2020).

In recent years, increasing interest has been shown in the use of hybrid models, which take advantage of multiple algorithms. For example, a Kalman filter is combined with dynamic data in a multilayer perceptrons (MLP) model (Chen et al. (2004), Yu et al. (2010), Zaki et al. (2013), Tantawy and Zorkany (2014)). In Yang et al. (2016), a genetic algorithm (GA) is combined with a support vector machine model to tune hyperparameters. Such hybrid models are shown to increase the accuracy of machine learning models.

From recent researches on bus travel time prediction, LSTM has often been selected to improve the accuracy and performance of the prediction model (Pang et al. (2018), Petersen et al. (2019), Han et al. (2020)). However, a Kalman filter has not yet been combined with LSTM in the literature.

Therefore, we summarize the contributions of this work as follows.

- We develop a new hybrid model taking a Kalman filter in an LSTM model and compare this with the hybrid model of Chen et al. (2004), which combines a Kalman filter with an MLP model, to see the prediction accuracy.
- We apply a GA to find a perfect combination set of hyperparameters for constructing both neural networks, i.e. number of neurons, hidden layers, batch size and time window size.
- We apply our hybrid models to predict the bus travel time for a bus line of the HOCHBAHN bus company in Germany based on the data of that bus line in 2019.
- We provide a recommendation for the bus companies to create better plans for vehicles.

In the following sections, we will describe our hybrid model in Section 2 and some computational results and a recommendation for the bus company in Section 3 along with the conclusion in Section 4.

2. Methods

The bus travel time model is constructed in two main phases as illustrated in Figure 1. In the first phase, the selected independent variables based on historical data of trips and weather data, namely day of week, time of day, trip and precipitation, are trained in the LSTM network with determined weight vectors to minimize the mean square error ($MSE = \frac{\sum_{j=1}^{P}\sum_{i=1}^{n}(y_{ij}-\hat{y_{ij}})^2}{P \cdot n}$, where y and $\hat{y}$ are the actual and predicted driving time, P is a number of output neurons and n is the sample size of the data). All independent variables are transformed with one-hot encoding while the travel time variable is normalized with the min-max scaling method. The hyperbolic tangent is selected to be an activation function of the neurons in the hidden layers of LSTM, while the ReLU function is employed with neurons on the last hidden layer. The sigmoid function is used for the recurrent procedure. The training data are passed through backpropagation through training 50 epochs with a 0.001 learning rate. The training process is terminated if there is no improvement of loss function after five epochs. We also apply a dropout technique to prevent overfitting on training the model with a rate of 0.2. A truncated Gaussian distribution is utilized to generate random initial weights and bias values, and every parameter is updated with the Adam optimizer. A similar implementation of LSTM can be found in Petersen et al. (2019) and Han et al. (2020).

Furthermore, the parameters in Table 1, including the number of neurons in the hidden layers, number of hidden layers, batch size and time window size, are tuned with GA to find the best combination. The initial settings of the number of neurons, number of hidden layers, batch size, and time window size are 16, 1, 32 and the length of the

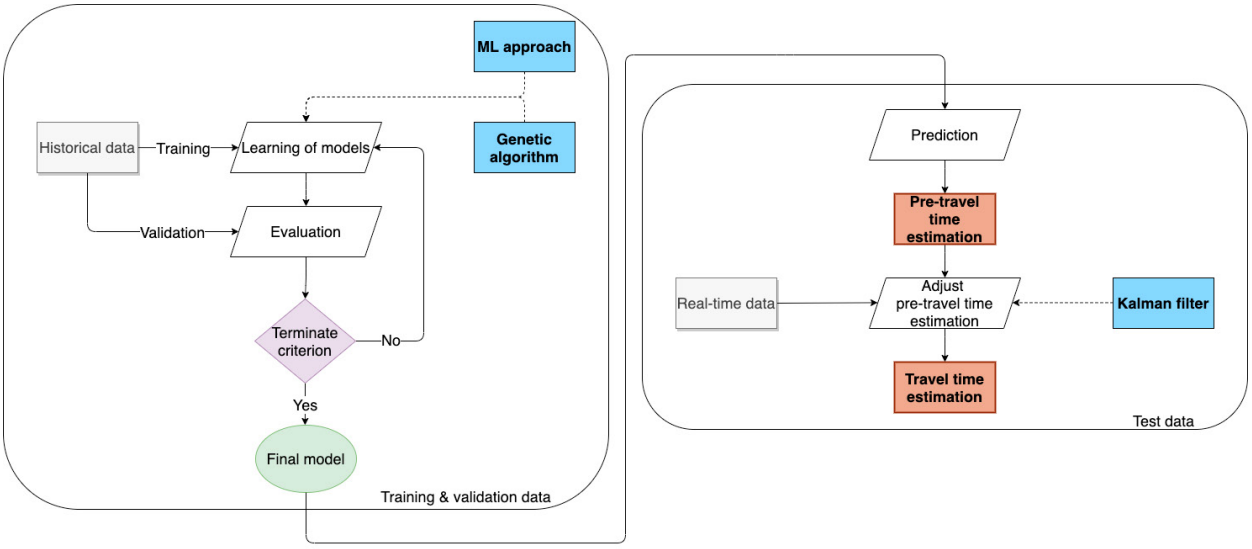

Fig. 1: The procedure for predicting bus travel time. "ML approach" is either the LSTM or the MLP model.

journey or the number of trips in a journey, respectively for both the MLP and LSTM approaches. Note that we define a journey as the travel from the beginning bus stop to the ending one, while a trip is the travel between two bus stops. More details about tuning using the GA can be found in Yang et al. (2016). The combinations in vectors establish a population. Each generation contains ten individual vectors that will be evaluated by the square root of MSE, a.k.a. the fitness function and then ranking the vectors. The four best combination sets are kept to become part of the next generation, while the other six disqualified vectors are arbitrarily picked with a 10% chance of being one part of the next generation. The free space of the new population is fulfilled with a new set of candidates, which the rest of the parents in this generation will create offspring by a cross-over method in which a child will get some bit-strings or parameters from one parent and retrieve another part from another parent. And there is a 20% chance that a mutation happens to the children to ensure genetic diversity. The population in this new generation runs through the same procedures of evaluation, ranking, offspring and mutation, and then repeats until there are five generations. Subsequently, the best combination of hyperparameters set in the last generation is implemented in the final neural network. A similar procedure on training the neural network and tuning the hyperparameters with GA is also applied to an MLP model as our baseline model for comparison.

Hyperparameter type	**Tuning set**
Number of neurons	[16, 32, 64, 128, 256]
Number of hidden layers	[1, 2, 3]
Regularization: Mini-batch	[16, 32, 64, 128, 256, 512]
Time window size for LSTM	[0.5l, 1l, 1.5l, 2l] where l is the number of trips of each journey

Table 1: The scope of hyperparameters for tuning.

In the second phase, the prediction from the first stage is used as a pre-travel time estimation. We use a Kalman filter with real-time data to adjust the pre-estimation for the arrival time of the next stop in this journey. The Kalman filter procedures can be separated into two parts: predict and update (see Figure 2). The predictor equations in the predict stage are used for estimating the current state and error covariance. $\hat{x}_t$ denotes the estimated state while P_t represents the estimated error covariance at time interval t. The state $\hat{x}_t$ has two dimensional variables of J_t and V_t which denote the duration between the stop $t-1$ and t and the arrival time up to the last stop respectively. In the initial state, $t = 0$, the predicted state $\hat{x}_0 = \binom{J_0}{V_0} = \binom{0}{0}$ and the covariance P_t is set to be $P_0 = \left(\begin{smallmatrix} 1 & 1 \end{smallmatrix}\right)$. The superscript $^-$ implies the output parameters from the prediction stage. The weight matrix A is a transition parameter to link between the

previous time step $t-1$ and the current step t, and describes the relationship between them; $A = \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right)$ is the initial setting. The input parameter u_t is the trip travel time from the pre-travel time estimation between stop $t-1$ and stop t that has matrix B to connect this input term with the state x. The matrix B is predefined as $B = \left(\begin{smallmatrix}-1\\ 1\end{smallmatrix}\right)$. The updater equations deal with improving the estimate values by an actual measurement including the actual driving time at time interval t, z_t. The weight matrix H links the state x_t to the measurement z_t with a predefined value $H = \left(\begin{smallmatrix}0 & 1\end{smallmatrix}\right)$. This stage introduces a weight matrix, namely *Kalman gain*, which plays a crucial role in relating the measurement and the estimation (Bishop, 2006). The way to combine Kalman with an MLP model can be found in Chen et al. (2004). In such a way, the prediction of the models based on historical and real-time data is expected to achieve higher accuracy (see next section).

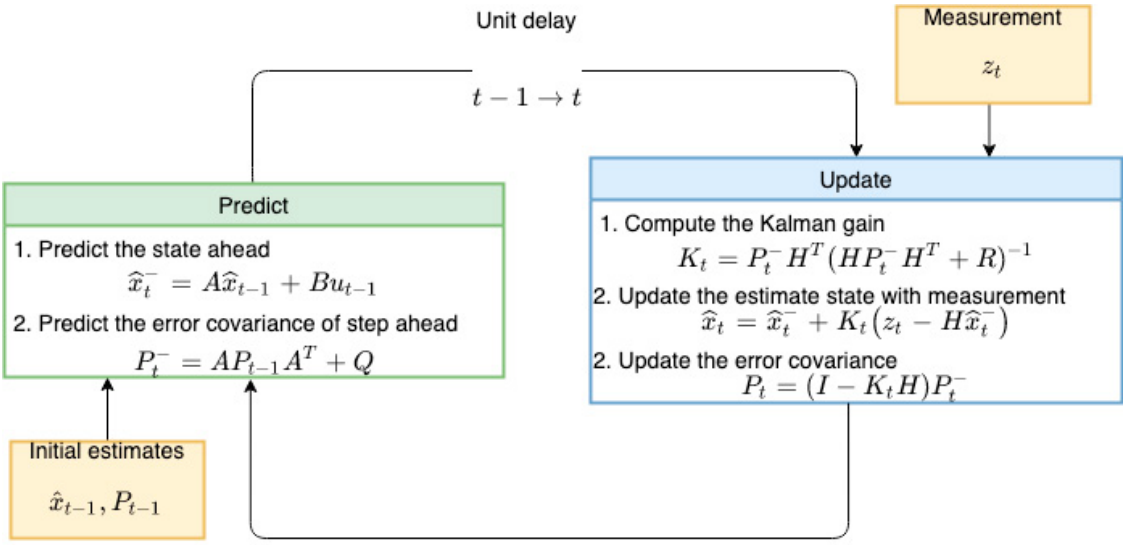

Fig. 2: Kalman filter cycle based on the study of Bishop (2006).

3. Computational results

The bus line data of our case study include four journey patterns, which contain 12, 10, 22 and 20 bus stops. We tested our models month by month in 2019 (except December) due to the limited computational power. We split the preprocessed data for each month into three parts for each journey pattern, so the data of the first two weeks are used for training, the data for the following week are used for validation, and we predict the bus travel time for the last week. Note that in our experiment we ignore holidays; bus service types "deadhead", "inbound" and "outbound"; and journeys with unfinished drives.

We use the same independent variables as in Chen et al. (2004) in our models, including the day of the week, time of the day, trip and precipitation. We also do a data analysis to see how they influence the actual driving times compared with the planned ones. It shows the following similar results for all journey patterns: the actual bus driving time on weekdays tends to be longer for each trip than on the weekends; the time of the day impacts the driving time, especially in the rush hours. The difference in the driving time for a trip can be up to e.g. 800 seconds at different times of the day. Lastly, the bus travel time tends to increase slightly during precipitation (snow/rain).

We forecast the bus arrival time for journey n, A_n, which is calculated using the sum of the bus travel times of all trips in n by considering independent variables. The objective of the model is to make the forecasted A_n as close as possible to the actual driving time, z_n, for each journey n. We use the $\text{RMSE}_n = \sqrt{\frac{\sum_{i=1}^{n}(z_i - A_i)^2}{n}}$ to measure the difference between A_n and z_n for each journey n. In other words, the model aims at minimizing RMSE_n for each journey n.

3.1. Test scenarios

In order to see the performance of our hybrid models, we test three scenarios. The prediction models are tested in these scenarios to determine which approach consistently outperforms the others. The *baseline* model, which is based on Chen et al. (2004), is denoted as MLP_KF. The hybrid models of neural network with GA are MLP_GA and LSTM_GA for MLP and LSTM respectively, while the hybrid models of neural network with GA and the Kalman filter are denoted as MLP_GA_KF and LSTM_GA_KF.

First, we want to see whether our hybrid models generally work well for all journey patterns. Figure 3 shows the results for one journey pattern with 12 stops (in other words, 11 trips) in April, and we can achieve similar results for all patterns in the same month. It is obvious that an approach using hybrid models of a neural network (either MLP or LSTM) with GA and the Kalman filter can achieve better accuracy than the baseline model, MLP_KF, especially the hybrid MLP_GA_KF model. Table 2 shows the percentage improvement of the RMSE for each hybrid model compared with the planned and baseline models respectively. The *planned* refers to the planned bus driving time from the bus company. Overall, MLP_GA_KF has a better improvement over the baseline model, with an improvement of over 35% in RMSE. Meanwhile, LSTM_GA_KF attains an improvement of over 28%. Additionally, we can conclude that the models without adjusting with real-time information (MLP_GA and LSTM_GA) tend to have a worse performance in all patterns. But these models still produce better results than the planned model. The non-hybrid models (MLP and LSTM) provide worse results than the planned model.

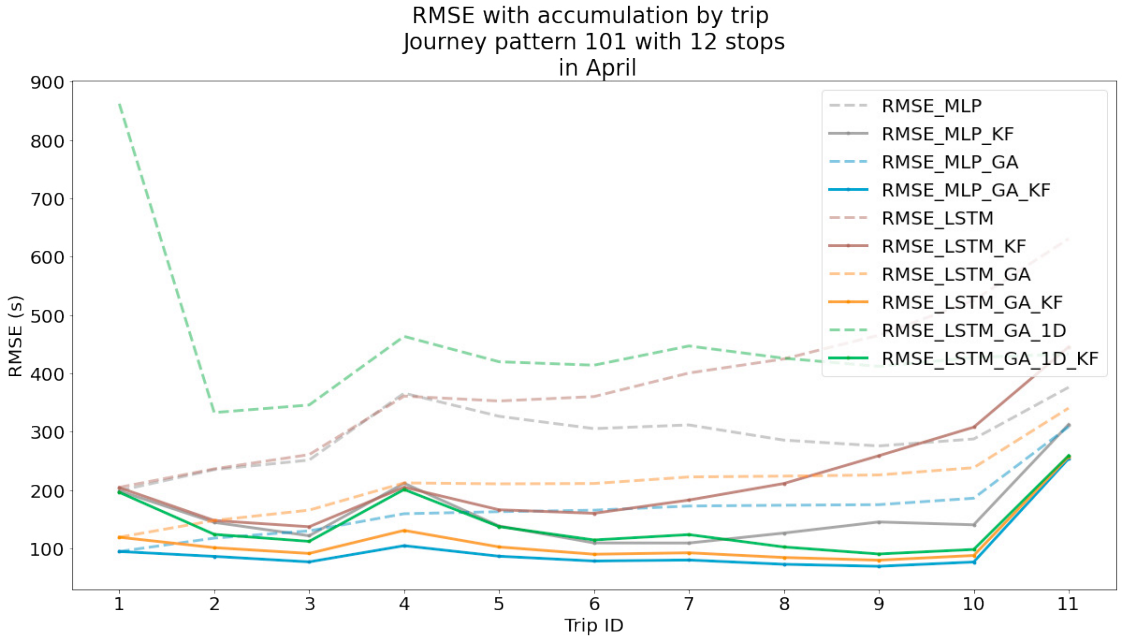

Fig. 3: Comparison of RMSEs for different models of journey pattern with 12 stops in April.

In the second scenario, we test our hybrid models for 11 months on the journey with 12 bus stops to examine the models consistently in different months related to seasonal factors. Similar results can be achieved as in the first scenario. The overall results in Table 3 show that MLP_GA_KF and LSTM_GA_KF can achieve higher accuracy than the baseline model, on average approximately 46% and 39% respectively. And their results are much better than the planned driving times from the bus company, on average 56.4% and 50.0% respectively. However, MLP_GA and LSTM_GA do not consistently provide better prediction for the whole year than the planned model, for example in January and February.

Lastly, the third experiment is a comparison between the performance of LSTM models for a univariate time series forecast (LSTM_GA_1D_KF, using only historical travel time), and the LSTM model for multivariate time series data, which takes four input factors into account (LSTM_GA_KF). Table 3 show that both models have a higher arrival time

Measurement	Model	Journey			
		12 stops	10 stops	22 stops	20 stops
RMSE (seconds)	Planned	236.70	124.78	390.46	190.38
	MLP	296.89	156.28	386.60	257.39
	MLP_KF (baseline)	169.78	110.29	205.28	167.60
	MLP_GA	175.57	87.75	218.26	135.36
	MLP_GA_KF	110.19	66.27	106.79	98.66
	LSTM	402.34	155.63	335.66	134.30
	LSTM_GA	217.49	97.88	269.73	127.59
	LSTM_GA_KF	122.22	71.17	131.20	93.77
Percent improvement: vs. Planned	MLP_KF	28.27%	11.61%	47.43%	11.96%
	MLP_GA	25.83%	29.68%	44.10%	28.90%
	MLP_GA_KF	53.45%	46.89%	72.65%	48.18%
	LSTM_GA	8.12%	21.56%	30.92%	32.98%
	LSTM_GA_KF	48.36%	42.97%	66.40%	50.75%
Percent improvement: vs. MLP_KF	MLP_GA_KF	35.10%	39.91%	47.98%	41.14%
	LSTM_GA_KF	28.01%	35.47%	36.09%	44.06%

Table 2: Percentage improvement from the planned and baseline models for all journeys, during April 2019.

Measurement	Model	Month										
		Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov
RMSE (seconds)	Planned	71.20	70.70	153.80	236.70	189.80	229.80	153.80	143.10	178.20	508.30	236.40
	MLP_KF (baseline)	97.37	97.03	132.43	169.86	140.71	148.98	127.50	115.61	132.62	273.13	159.83
	MLP_GA	78.60	57.90	130.30	175.60	132.10	168.80	145.10	104.70	148.40	377.60	118.00
	MLP_GA_KF	48.76	37.43	67.36	110.25	69.34	89.31	79.25	60.42	70.86	181.34	60.09
	LSTM_GA	83.5	74.1	155.6	217.5	138.0	167.8	144.6	102.4	215.5	398.6	149.5
	LSTM_GA_KF	49.53	48.70	89.58	122.36	70.31	87.53	81.48	58.48	120.01	182.69	68.32
	LSTM_GA_1D_KF	63.80	52.76	74.74	150.60	99.41	131.21	88.13	60.33	74.76	208.61	122.33
Percent improvement: vs. Planned	MLP_KF	-36.76%	-37.24%	13.89%	28.24%	25.87%	35.17%	17.10%	19.21%	25.58%	46.27%	32.39%
	MLP_GA	-10.34%	18.10%	15.30%	25.83%	30.39%	26.53%	5.64%	26.87%	16.75%	25.71%	50.10%
	MLP_GA_KF	31.52%	47.05%	56.20%	53.42%	63.47%	61.14%	48.47%	57.78%	60.24%	64.33%	74.58%
	LSTM_GA	-17.23%	-4.87%	-1.18%	8.12%	27.29%	26.97%	5.97%	28.47%	-20.94%	21.57%	36.75%
	LSTM_GA_KF	30.44%	31.12%	41.75%	48.31%	62.96%	61.91%	47.02%	59.14%	32.66%	64.06%	71.10%
	LSTM_GA_1D_KF	10.39%	25.38%	51.40%	36.38%	47.62%	42.90%	42.70%	57.84%	58.05%	58.96%	48.25%
Percent improvement: vs. MLP_KF	MLP_GA_KF	49.93%	61.42%	49.13%	35.09%	50.72%	40.05%	37.84%	47.74%	46.57%	33.61%	62.40%
	LSTM_GA_KF	49.14%	49.81%	32.35%	27.97%	50.03%	41.24%	36.10%	49.42%	9.51%	33.11%	57.26%
	LSTM_GA_1D_KF	34.48%	45.63%	43.56%	11.34%	29.35%	11.93%	30.87%	47.82%	43.63%	23.62%	23.46%

Table 3: Percentage improvement from the planned and baseline models for journey with 12 stops, from January 2019 to November 2019.

prediction accuracy for all months than the baseline model and planned model. Furthermore, LSTM_GA_KF provides better results than LSTM_GA_1D_KF compared with the baseline model (on average 39% and 31% better) and the planned model (on average 50% and 43.6% better). The exceptions are the results for July and September.

3.2. Hardware and frameworks

All travel time prediction models in this study are implemented in the Python programming language. The main Python libraries that are used in data preprocessing and model evaluation are Pandas, NumPy and scikit-learn. Keras 2.4.3, which is an API library framework built on top of TensorFlow 2.0 (Gulli and Pal, 2017), is used for developing neural network models. The training process is performed on Google Colaboratory (Colab) (Bisong, 2019). The hardware used for Colab was an NVIDIA Tesla T4 GPU with a CPU processor consisting of 2-core Xeon 2.2GHz and 13 GB of RAM. In the process of training with GA, the models ran for times varying between 0.5 and 25 hours. LSTM_GA took the longest time, around 2–25 hours to finish the process, while the MLP_GA models took around 0.5 and 4.5 hours and the LSTM_GA_1D took between 1 and 9 hours.

3.3. Recommendation for bus companies

As mentioned in the introduction, the precise prediction of bus travel time is important in the scheduling of buses by bus companies. In recent years, the increasing usage of e-buses has required more precise prediction of bus travel times due to the limited battery time. Moreover, when they are stuck in traffic jams, this also requires energy to supply the auxiliaries for e-buses due to, for example, heating in winter or air-conditioning on rainy days (see De Cauwer et al. (2015)). Based on the results in Subsection 3.1, we want to make a recommendation for bus companies to improve their planned bus travel times in the vehicle scheduling problem. The vehicle scheduling problem addresses the task of assigning buses to cover a given set of timetabled trips with consideration of practical requirements; an overview of this problem can be found in Bunte and Kliewer (2009). The result of this problem is a sequence of trips for each bus, from leaving its depot to going back to that depot. The result of our predicted driving time for each trip can be used as input to the vehicle scheduling problem, instead of the planned driving time.

However, due to the structure of the planning problem, the plan for buses should be estimated before a new month begins. That means that the real information cannot be considered during the planning, at least not yet. Therefore, our approach with the Kalmin filter is not suitable for this application. Instead, our combined models of both MLP and LSTM with GA can be used.

In order to see how well our hybrid models work for a larger training set, we use, for example, the May and June data for training, and the July data for validation. And we predict the bus travel times for August. We use the similar procedure for the travel time prediction as shown in Figure 1, but without adjusting pre-travel time estimation. We test our models on a PC with an Intel Xeon Gold 6139M CPU @ 2.3GHz 18 core (two threads per core) and 32GB of RAM, running on a Ubuntu Server 18.04 with 5 LTS. Table 4 shows the results of the hybrid models (MLP_GA and LSTM_GA) for all journey patterns and the percentage improvement compared with the results of the planned bus travel times from the bus company. In general, our hybrid models both provide similar results, namely 20%–36% improvement in the accuracy of the bus travel times. We also check the computational time of both models. MLP_GA needs on average less than 15% of the computational time needed by LSTM_GA. Note that the computational time to solve MLP_GA is on average about 6,100 seconds. The chosen parameters by GA for both models are listed in Table 5.

Measurement	**Model**	**Journey**			
		12 stops	**10 stops**	**22 stops**	**20 stops**
RMSE (seconds)	Planned	172.9	108.3	330.1	162.6
	MLP_GA	127.5 (26.2%)	81.3 (24.9%)	218.5 (33.8%)	129.9 (20.1%)
	LSTM_GA	124.7(27.9%)	83.0 (23.4%)	212.4 (35.7%)	123.3 (24.2%)

Table 4: The RMSE values of planned vs. hybrid models for the prediction of bus travel times in August. In brackets is the percentage improvement compared with the RMSE of the planned bus travel time.

Hyperparameter type	**MLP_GA**	**LSTM_GA**
Number of neurons	32	16
Number of hidden layers	3	3
Regularization: Mini-batch	[16, 32, 64]	32
Time window size for LSTM	-	2

Table 5: The tuned parameters in GA for the MLP_GA and LSTM_GA models.

Based on the computational results above, we want to make the following suggestion for bus companies. It is worth applying MLP with GA, since it provides better prediction of bus travel time without much more computational effort. Furthermore, different factors that might cause delays, such as construction sites or bad weather, are considered in MLP.

4. Conclusion

In this work, we introduce new hybrid models which combine a Kalman filter with an LSTM and an MLP model while applying a GA to find a good combination of parameters in the neural network to predict real-world bus travel times. On the basis of our experimental results, our implemented MLP combined with a Kalman filter and GA provided the best prediction. We can improve the prediction for each journey pattern by on average of 45.44% compared with the hybrid model we implemented as in Chen et al. (2004). Furthermore, based on the further computational results in Section 3.3, we recommend that bus companies take the predicted bus travel times provided by MLP_GA, rather than the planned times to solve the vehicle scheduling problem.

Acknowledgements

The authors want to greatly thank HOCHBAHN for providing us with data for the analysis. This work is a contribution to the "Robust integrated vehicle scheduling, crew scheduling and rostering in public bus transit" project, which is funded by the German Research Foundation grants KL 2152/7-1 and LX 156/2-1.

References

Agrawal, S., 2020. A survey of bus arrival time prediction methods, in: Security with Intelligent Computing and Big-Data Services 2019: Proceedings of the 3rd International Conference on Security with Intelligent Computing and Big-Data Services (SICBS), December 4–6, 2019, New Taipei City, Taiwan, Springer Nature. p. 197.

Altinkaya, M., Zontul, M., 2013. Urban bus arrival time prediction: A review of computational models. International Journal of Recent Technology and Engineering (IJRTE) 2, 164–169.

Bishop, C.M., 2006. Pattern recognition and machine learning. Springer.

Bisong, E., 2019. Google Colaboratory. Apress, Berkeley, CA. pp. 59–64. doi:10.1007/978-1-4842-4470-8_7.

Bunte, S., Kliewer, N., 2009. An overview on vehicle scheduling models. Public Transport 1, 299–317.

Chen, M., Liu, X., Xia, J., Chien, S.I., 2004. A dynamic bus-arrival time prediction model based on APC data. Computer-Aided Civil and Infrastructure Engineering 19, 364–376.

Choudhary, R., Khamparia, A., Gahier, A.K., 2016. Real time prediction of bus arrival time: A review, in: 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), IEEE. pp. 25–29.

De Cauwer, C., Van Mierlo, J., Coosemans, T., 2015. Energy consumption prediction for electric vehicles based on real-world data. Energies 8, 8573–8593.

Gulli, A., Pal, S., 2017. Deep learning with Keras. Packt Publishing Ltd.

Han, Q., Liu, K., Zeng, L., He, G., Ye, L., Li, F., 2020. A bus arrival time prediction method based on position calibration and LSTM. IEEE Access 8, 42372–42383.

Pang, J., Huang, J., Du, Y., Yu, H., Huang, Q., Yin, B., 2018. Learning to predict bus arrival time from heterogeneous measurements via recurrent neural network. IEEE Transactions on Intelligent Transportation Systems 20, 3283–3293.

Petersen, N.C., Rodrigues, F., Pereira, F.C., 2019. Multi-output bus travel time prediction with convolutional LSTM neural network. Expert Systems with Applications 120, 426–435.

Tantawy, M., Zorkany, M., 2014. A suitable approach for evaluating bus arrival time prediction techniques in Egypt. Algorithms 2, 9.

Yang, M., Chen, C., Wang, L., Yan, X., Zhou, L., 2016. Bus arrival time prediction using support vector machine with genetic algorithm. Neural Network World 26, 205.

Yu, B., Yang, Z.Z., Chen, K., Yu, B., 2010. Hybrid model for prediction of bus arrival times at next station. Journal of Advanced Transportation 44, 193–204.

Zaki, M., Ashour, I., Zorkany, M., Hesham, B., 2013. Online bus arrival time prediction using hybrid neural network and Kalman filter techniques. International Journal of Modern Engineering Research 3, 2035–2041.