Noname manuscript No.
(will be inserted by the editor)

Comparative Validation of Spatial Interpolation Methods for Traffic Density for Data-driven Travel-time Prediction

Hiroki Katayama · Shohei Yasuda · Takashi Fuse

Received: date / Accepted: date

Abstract In data-driven travel-time prediction, previous studies have mainly used speed as the input. However, from a traffic engineering perspective, given that speed varies little in the free-flow regime, traffic density, which can accurately represent traffic conditions from the free-flow regime to the congested-flow regime, is preferable as an input. In this study, we compared the accuracy of traffic densities spatially interpolated using spatial statistical and machine learning methods, and validated their effectiveness as inputs for travel-time prediction. The results show that even traffic density interpolated by simple spatial interpolation contributes to the accuracy of travel-time prediction and is superior to speed for early detection of traffic congestion.

Keywords Traffic density interpolation · Travel time prediction · Spatial interpolation

1 Introduction

With the development of intelligent transport systems (ITS), data-driven travel-time prediction methods for large-scale networks have been actively developed. It was estimated that the time lost due to traffic congestion is equivalent to approximately 2.8 million man-hours of labor per year in Japan [1]. Accurate travel-time prediction over a wide area is expected to contribute to reducing traffic congestion through traffic control by road administrators, for example by regulating the traffic flow into congested sections and helping drivers avoid congestion [2]. Conventional travel-time prediction is based on simulations using mathematical models based on the traffic-flow theory; however, it has been pointed out that it is difficult to predict traffic conditions on large networks [3][4]. By contrast, data-driven methods have shown that prediction accuracy does not change significantly as networks become larger [5]. Furthermore, data-driven travel-time prediction methods have been actively developed owing to the accumulation of wide-area and spatially continuous traffic data observed by the widespread use of global navigation satellite system (GNSS) equipment and the improvement of computer performance.

In most previous studies, time-series data on speed were used as the traffic state variable input for data-driven travel-time prediction [6]. From a traffic engineering perspective, traffic density, which can accurately represent traffic conditions from the free-flow regime to the congested-flow regime, is preferable as an input to speed, which changes less and has high variance in the free-flow regime. While speed does not change significantly until traffic congestion occurs, traffic density changes with the number of vehicles even when there is no traffic congestion. Therefore, it is expected that using traffic density as an input will allow more accurate detection of traffic congestion. According to a review of travel-time prediction using data-driven methods from 2010 to 2021, only three out of 115 studies used density as an input, and even when density was used as an input, the prediction target was limited to a single road [6]. This is because it is difficult to obtain wide-area observation data for traffic density because of the high maintenance cost of observation equipment,

H. Katayama
Department of Civil Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113–8656, Japan
E-mail: katayama@trip.t.u-tokyo.ac.jp

S. Yasuda
E-mail: s.yasuda@civil.t.u-tokyo.ac.jp

F. Takashi
E-mail: fuse@civil.t.u-tokyo.ac.jp

whereas it is easy to obtain wide-area observation data for speed through mobile observations using GNSS.

Given that there is a correlation between speed and traffic density, it is expected that traffic density can be spatially interpolated by combining the spatially sparse traffic density observed by detectors and the spatially dense speed observed by GNSS. Although there have been studies estimating daily traffic volume using spatial statistical methods [7] and interpolating on GNSS and detector data for single roads, no study has estimated hourly traffic density for large-scale networks.

Therefore, in this study, using the density observed by detectors and the speed observed by GNSS, we performed spatial interpolation of traffic density using methods based on spatial statistics and machine learning, and compared and validated the accuracy of the interpolation. Furthermore, travel-time prediction was performed using interpolated densities to confirm their validity. Because direct observation of traffic density is difficult in practical terms, we conducted microscopic simulations in which the density was observable for validation.

2 Methods

This section describes the method used for spatial interpolation of traffic density and the method used for travel-time prediction to validate the difference between observed and interpolated traffic density.

2.1 Density interpolation method

In this study, a road network was represented as a graph. Specifically, intersections, branches, mergers, and dead ends were considered as nodes, and roads connecting the nodes were considered as edges. The target of the interpolation was all edges of the graph. However, each interpolation method described below estimates the density at a point; therefore, a representative point was set at an edge, and the interpolated density at the representative point was used as the interpolated density of the edge. In this study, the midpoint of the edge was set as the representative point. The target of the interpolation was edges whose density was unknown, and the density interpolation was performed at each time step.

Four density interpolation methods were considered: inverse distance weighting (IDW), ordinary kriging (OK) [8], cokriging, and extreme gradient boosting (XGBoost) [9]. IDW, OK, and cokriging are based on Tobler's first law of geography [10], which states that the relationship between objects that are close to each other increases with distance. XGBoost, by contrast, is a machine learning method that uses decision trees. IDW is a method for interpolating the values of unknown points using a weighted average of weights that are the powers of the inverse of the distance between unknown points and known points. Compared with OK and cokriging, IDW has simpler weights and lower computational complexity, but it does not consider the relative positions of the observed points. Compared with Kriging and cokriging, IDW has lower computational complexity owing to the simplicity of the weights, but it dose not consider the relative positions of the observation points. OK also interpolates the value of unknown points using a weighted average of the surrounding observations. First, the covariance of the error is modeled as a function of the distance from the observed values of known points. Next, the expected value of the variance is calculated on the basis of the covariance estimated from the covariance function, and the weights that minimize this expected value are calculated. Next, the expected value of the variance is calculated using the covariance estimated from the covariance function, and the weights that minimize this expected value are calculated. Although the computational complexity of kriging is greater than that of IDW, it considers the relative positions of the observed value points. It can also calculate the prediction error [11]. cokriging is a kriging method with an auxiliary variable that is correlated with the target variable. cokriging is effective when the observed values of the predictor are sparsely available; however, the observed values of another variable that is correlated with the predictor are densely available. XGBoost is a machine-learning method that uses gradient boosting to weight weak learners based on decision trees. XGBoost achieves high accuracy in interpolating spatially correlated targets by using latitude and longitude as inputs [12][13]. XGBoost is characterized by its ability to add a variety of variables compared with the previous three methods. The coordinates of all edges and the densities of edges whose densities are known are used as inputs for IDW and OK. The coordinates and velocities of all edges and the densities of edges with known densities are used as inputs for cokriging and XGBoost.

2.2 Travel-time prediction method

The densities interpolated in the previous section were used to make travel-time predictions for all edges after a few time steps. Given that the edge distances are known, the travel time is calculated by dividing the distance by the velocity. Therefore, velocity was used

as the prediction target. We used GCN-LSTM, a composite model of graph convolutional network (GCN) and long short-term memory (LSTM), to validate the accuracy of the travel-time prediction. GCN extracts the spatial correlations of the input, whereas LSTM extracts the temporal correlations of the input. This model is available in the Python StellarGraph package [14].

3 Validation of spatial interpolation

The accuracy of the density interpolation was validated using data generated by microsimulation. Because density is difficult to observe in actual traffic, we generated data for validation using a simulation that can observe the density of the entire network. The ability to observe the density over the entire network allows us to compare and validate the accuracy of travel-time predictions when the input is the interpolated density and also when the input is the observed density.

3.1 Simulation overview

The true values of the density and velocity for validation were generated using microscopic simulations. We used Simulation of Urban Mobility (SUMO), an open-source micro traffic simulator developed by the German Aerospace Center. SUMO takes the network and demand as inputs to the simulation and provides density and velocity values for a cross section or region.

Network data for SUMO consist of nodes, which correspond to intersections, and edges, which correspond to roads. Nodes store information, such as position coordinates and traffic signal control. Edges store information, such as adjacencies to nodes, number of lanes, and speed limits. The network used in this study is a simplified network of roads in Sioux Falls, South Dakota, USA [15]. These network data are widely used as validation data in transportation engineering research [16][17].

3.1.1 Input of simulation

SUMO requires information such as the connection relationship between nodes and edges, location coordinates of nodes, lanes within an intersection, traffic signal control, number of lanes at an edge, and the speed limit. These parameters are described in this section, but SUMO's default settings are retained for settings that are not described here.

The latitude and longitude of the nodes, edge adjacencies, and demand between nodes were obtained from data published by Transportation Networks for Research [18], with some modifications. The demand was given as the number of vehicles moving between arbitrary nodes per day; however, in order to reproduce daily and weekly changes in traffic volume in the simulation, the demand between nodes was varied hourly. Specifically, cross-sectional traffic volume data for arterial roads in Hyogo Prefecture in October 2021, published by the Japan Road Traffic Information Center, were aggregated on an hourly basis to obtain the ratio of traffic volume to daily traffic volume. The demand between nodes was calculated based on this ratio, rounded down to the nearest whole number.

Because SUMO does not allow vehicles to be generated and extinguished within an intersection, vehicles were generated and extinguished at the virtual edges near the intersection. Specifically, edges of approximately 30 m were set up near the intersection and connected to all the edges flowing into and out of the intersection. Vehicles that had a destination at the intersection diverged before the intersection and disappeared toward the virtual edges. By contrast, vehicles originating at the intersection were generated at the virtual edge and merged immediately after the intersection. The vehicle settings were not changed from their default values. Vehicles were randomly generated and traveled through the shortest route from the starting point to the destination. Traffic signals were installed at intersections without expressway connections. The number of lanes and speed limits were obtained using Google Maps Street View. The intersection settings vary depending on whether an expressway is connected to an intersection. All intersections with expressway connections were set up as interchanges to match real roads in Sioux Falls. The intersection turn settings were not changed from the default settings, except that U-turns were prohibited at all intersections.

3.1.2 Output of simulation

The simulations yielded true values of density and velocity for subsequent validation. The network was divided into 300-m edges for each edge, and the density and velocity were obtained every 5 min for all 588 edges for one month from 00:00-04:59 on October 1, 2021, to 23:55-23:59 on October 31, 2021. The network was divided to validate the accuracy of the density interpolation by varying the intervals between the edges whose densities were known. However, because it is impossible to divide all the edges into 300-m edges, each edge was divided so that there are the greatest possible number of 300-m edges under the condition that the edges at

both ends after the division had the same length and were longer than 150 m.

3.2 Validation

This section describes the conditions for data preprocessing and density interpolation validation. In the simulation, edges where no vehicles passed during a 5-min period had missing values for density and speed. Missing values for density were set to zero, whereas missing values for speed were linearly interpolated in the time direction. However, if the period ended with a missing value, it was replaced by the end of the normally observed value. Given that kriging interpolates the density and velocity after logarithmic transformation, 0.01 was added to the density and velocity to prevent them from diverging after the transformation. The densities and velocities with 0.01 added were treated as true values, and the accuracy was validated using these values.

The density interpolation was validated by varying the detector installation interval, and the interpolation accuracy was validated with five detector installation patterns: 1, 3, 5, 7, and 9 edges apart. Each pattern corresponds to spacings of 600 m, 1200 m, 1800 m, 2400 m, and 3000 m, respectively, given that the edges were divided into approximately 300-m sections. To quantitatively investigate the effect of the detector installation intervals on the interpolation accuracy, the detectors were installed at equally spaced intervals. Because it is difficult to install detectors equally spaced throughout the entire network, they were installed equally spaced within the edge before segmentation. Specifically, when the installation interval was , a detector was installed on the (+ 1)-th post-division edge counting from the starting point of the pre-division edge, and detectors were installed every edges thereafter until the end point. Density interpolation was applied to the edges where no detector was installed when the detector installation interval was 2. This is because no detectors were installed at these edges in the other detector installation patterns (Fig. 1).

To consider the influence of the time of day, 24 patterns were tested for each method and detector installation pattern. Specifically, the time period from 00:00 to 05:00 for each hour was selected as representative of each hour. The data on October 1, 2021 were used for the validation.

The density interpolation methods used in this study were IDW, OK, cokriging, and XGBoost. IDW, OK, and cokriging use the gstat package[19] in R, whereas XGBoost uses the xgboost package in Python. Concerning IDW, the inverse square of the distance was used as the weight. The inputs to XGBoost were

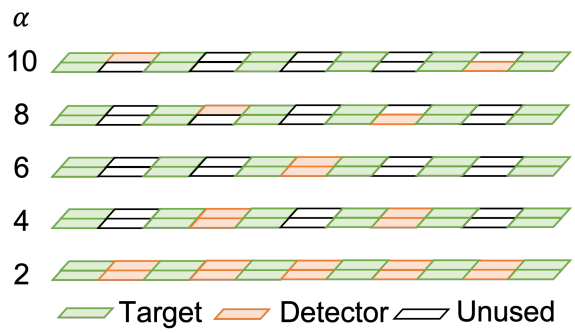

Fig. 1 Detector installation pattern

the velocity, latitude, and longitude, and all parameters were set to default values. In OK, the density was log-transformed and used as an input to make the right-skewed density distribution more symmetrical. In cokriging, log-transformed densities and velocities were used as inputs. This is because the correlation coefficient between the density and velocity after logarithmic transformation was higher than that before logarithmic transformation, in addition to making the right-skewed density distribution as symmetrical as that of OK. Given that OK and cokriging require that the density and velocity at the same coordinate have the same value, when there were multiple detectors at the same coordinate, the average of the density and harmonic average of the velocity were taken as the representative values at that coordinate. The mean absolute error (MAE) and root mean square error (RMSE) were used as evaluation metrics for interpolation accuracy. RMSE and MAE are expressed as

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{1}$$

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|; \tag{2}$$

where $\hat{y}_i$ denotes the predicted value of sample i, y_i denotes the true value, and N denotes the number of samples. We interpolated 20 patterns with different installation intervals of detectors and methods as described above.

3.3 Results

The interpolation accuracies aggregated over all edges and the time periods for each detector installation interval are listed in Table 1. From Table 1, it can be observed that both MAE and RMSE tend to increase as

Table 1 Accuracy of density interpolation by detector interval

	metrics	cokriging	IDW	OK	XGBoost
2	MAE	1.6774	1.5726	**1.5360**	1.6531
	RMSE	5.8695	5.7030	**5.5885**	6.1611
4	MAE	1.6994	**1.6077**	1.7033	1.6629
	RMSE	**5.6850**	5.8477	5.7850	5.9027
6	MAE	2.0949	1.9096	**1.8333**	2.2675
	RMSE	6.2415	6.4360	**5.8748**	6.2780
8	MAE	**1.8506**	2.1477	2.2157	2.1898
	RMSE	**5.9688**	6.3955	6.3639	6.3576
10	MAE	**2.0214**	2.3460	2.2758	2.6960
	RMSE	**6.2155**	6.6880	6.5648	6.9212

Table 2 Accuracy of density interpolation by method

methods	cokriging	IDW	OK	XGBoost
MAE	**1.8687**	1.9167	1.9128	2.0939
RMSE	**5.9997**	6.2253	6.0466	6.3330

the detector interval increases. The difference in accuracy between the methods also increases as the detector interval increases. The method with the best interpolation accuracy varies with the detector interval. For narrower detector intervals, such as 2, 4, and 6, OK, cokriging, and IDW were the most accurate, whereas for wider detector intervals such as 8 and 10, cokriging was the most accurate.

The interpolation accuracies aggregated over all time periods and all detector placement patterns are listed in Table 2. According to Table 2, cokriging has the best accuracy in terms of both MAE and RMSE. The difference in accuracy between cokriging and OK was as small as approximately 0.05 for both MAE and RMSE, indicating that OK is comparable to cokriging in terms of interpolation accuracy. The same is true for IDW.

4 Validation of travel time prediction

The interpolated densities were used to predict travel-time, and whether the interpolated densities were accurate enough for travel-time prediction was validated by comparing them to the case in which the true densities or velocities were used as inputs for travel-time prediction.

Travel-time predictions were made using GCN-LSTM with different inputs such as speed, true density, and interpolated density. The period from 00:00, October 1, 2021, to 23:59, October 24, 2021, was used for training, and the period from 00:00, October 25, 2021, to 23:59, October 31, 2021, was used for validation.

Using the last 10 time-steps as input, we predicted the velocity at all edges after 12 time-steps; namely, we predicted the velocity one hour later based on the last 50 min of information. Therefore, a set of input and correct values was created with a width of 22 time-steps. The sets were created within the training and validation periods, shifting by one time step in each case. The 6891 sets created within the training period were used as the training data, whereas the 1995 sets created within the validation period were used as the validation data.

Although cokriging was the most accurate density interpolation method tested in the previous section, the difference in accuracy between OK and IDW was small. To select the interpolation method to use, we also considered the computational complexity of each method. Given that OK and cokriging require the inverse of the covariance matrix between the observed value points for each interpolated point, the time complexity of OK and cokriging is $\mathcal{O}(N_\mathrm{p} N_\mathrm{o}^3)$, whereas that of IDW is $\mathcal{O}(N_\mathrm{p} N_\mathrm{o})$, where N_o is the number of observation points and N_p is the number of points to be interpolated. Therefore, in this study, densities interpolated using IDW, which are computationally inexpensive and can be expected to provide high interpolation accuracy, were used as inputs for travel-time prediction. The input was normalized by $[0; 1]$.

The hyperparameters of GCN-LSTM were set to 16 and 10 for the dimension of features that each node has in the output of the first and second layers of GCN, respectively, and 400 for the dimension of the LSTM output in both the first and second layers. The dropout ratio was set to 0.5. The optimization algorithm was Adam [20], the loss function was MAE, the number of training epochs was 200, and the batch size was 60.

MAE, RMSE, and MAPE were used as metrics for accuracy evaluation. MAPE is expressed as

$$\mathrm{MAPE} = \frac{100}{N} \sum_{i=1}^{N} \frac{| y_i - \hat{y}_i |}{y_i}, \tag{3}$$

where $\hat{y}_i$ denotes the predicted value of sample i, y_i denotes the true value and N denotes the number of samples.

4.1 Results

The accuracy of the travel-time predictions, validated with three patterns of input, namely, true density, interpolated density, and speed, is shown in Table 3. Regarding MAE and MAPE, the highest accuracy was achieved when the interpolated density was used, and for RMSE, the highest accuracy was achieved when the observed density was used.

Table 3 Accuracy of travel-time prediction by input

metrics	density (true)	density (interpolated)	speed
MAE	2.2164	**2.2107**	2.2753
MAPE	3.3421	**3.3370**	3.4333
RMSE	**3.9937**	4.0076	4.2902

The difference between the accuracy when the interpolated density was used as input and the accuracy when the true density was used as input is very small, less than 0.01 for both indices; therefore, it can be assumed that there is no significant difference in accuracy. By contrast, compared to the case in which velocity was used as input, the prediction accuracy improved by approximately 0.06 for MAE, 0.1 for MAPE, and 0.3 for RMSE when interpolated density or true density was used as input.

Next, we compared and validated the differences in the prediction results for the three patterns: 1) using the interpolated density as the input; 2) using the observed density as the input; and 3) using the velocity as the input. Fig. 2 plots the predictions of the three patterns and the correct answers for the five edges with the largest changes in velocity for the first day of the validation period. The change in velocity was defined as the difference between the 20-th and 80-th percentile values of the velocity.

In the free-flow regime, there is little difference in the prediction results as a function of the input; however, in the congested region, the prediction results change as a function of the input. Focusing on the sharp drop in speed between 7 a.m. and 8 a.m., i.e., during the occurrence of traffic congestion, the speed drop can be predicted when the true density is used as input or when the interpolated density is used as input, whereas when speed is used as input, the occurrence of traffic congestion is predicted with a delay of one hour, and the occurrence of traffic congestion is not predicted.

5 Discussion and Conclusion

To use traffic density as an input for travel-time prediction, we compared and validated the accuracy of spatial interpolation among four methods: IDW, OK, cokriging, and XGBoost. The results show that the difference in accuracy between the methods was small when the detectors were densely placed; however, the accuracy of cokriging was relatively better when the detectors were sparsely placed. This may be due to the fact that cokriging interpolates the density using not only the traffic density observed at the detectors, but also the correlation between the density and the speed observed throughout the network.

A comparison of travel-time prediction using the true traffic density as input and the traffic density interpolated by the IDW as input show no significant difference in accuracy. This indicates that even traffic density interpolated by the simple IDW method can be interpolated with sufficient accuracy to be used as input for travel-time prediction.

The results of travel-time prediction using speed as input and traffic density as input show that the accuracy of prediction is better when density is used as input. In particular, the prediction accuracy is better when traffic congestion occurs. This may be due to the fact that the density can accurately represent traffic conditions even in free-flow conditions, while speeds do not change significantly in free-flow conditions, and thus patterns that lead to congestion can be learned more efficiently.

To validate the accuracy of interpolating densities and the difference in accuracy of travel-time prediction between interpolated traffic density as input and true density observed throughout the network, experiments were conducted using simulation-generated data. However, the simulation-generated data were qualitatively different from the observed data of real traffic conditions; therefore, validation using actual observed data remains as a future challenge.

Acknowledgements This work was supported by JSPS KAKENHI Grant Number 21K14266.

Conflict of interest

The authors declare that they have no conflict of interest.

References

1. T. Ministry of Land, Infrastructure, T. of Japan. White paper on national capital region development (2017)
2. N. Ryo, Y. Shohei, I. Takamasa, Y. Asakura, Journal of Japan Society of Civil Engineers, Ser. D3 (Infrastructure Planning and Management) **70**(5), I_971 (2014)
3. C.F. Daganzo, Transportation Science **32**(1), 3 (1998)
4. C.F. Daganzo, Transportation Research Part B: Methodological **41**(1), 49 (2007)
5. S. Oh, Y.J. Byon, K. Jang, H. Yeo, Transport Reviews **35**(1), 4 (2015)
6. A. Abdi, C. Amrit, PeerJ Computer Science **7**, e689 (2021)
7. B. Selby, K.M. Kockelman, Journal of Transport Geography **29**, 24 (2013)
8. D.G. Krige, Journal of the Southern African Institute of Mining and Metallurgy **52**(6), 119 (1951)
9. T. Chen, C. Guestrin, in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (2016), pp. 785–794