applied sciences

MDPI

Article

Travel Time Prediction on Long-Distance Road Segments in Thailand

Rathachai Chawuthai [1,*], Nachaphat Ainthong [1], Surasee Intarawart [1], Niracha Boonyanaet [1] and Agachai Sumalee [2]

1 School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand; 61010279@kmitl.ac.th (N.A.); 61011432@kmitl.ac.th (S.I.); 61010588@kmitl.ac.th (N.B.)
2 School of Integrated Innovation, Chulalongkorn University, Bangkok 10330, Thailand; agachai.s@chula.ac.th
* Correspondence: rathachai.ch@kmitl.ac.th; Tel.: +662-329-8341 (ext. 114)

Abstract: This study proposes a method by which to predict the travel time of vehicles on long-distance road segments in Thailand. We adopted the Self-Attention Long Short-Term Memory (SA-LSTM) model with a Butterworth low-pass filter to predict the travel time on each road segment using historical data from the Global Positioning System (GPS) tracking of trucks in Thailand. As a result, our prediction method gave a Mean Absolute Error (MAE) of 12.15 min per 100 km, whereas the MAE of the baseline was 27.12 min. As we can estimate the travel time of vehicles with a lower error, our method is an effective way to shape a data-driven smart city in terms of predictive mobility.

Keywords: GPS data analytics; machine learning; smart mobility; travel time prediction

check for updates

Citation: Chawuthai, R.; Ainthong, N.; Intarawart, S.; Boonyanaet, N.; Sumalee, A. Travel Time Prediction on Long-Distance Road Segments in Thailand. *Appl. Sci.* **2022**, *12*, 5681. https://doi.org/10.3390/app12115681

Academic Editor: Dongsoo Har

Received: 18 April 2022
Accepted: 29 May 2022
Published: 2 June 2022

1. Introduction

For data-driven smart cities, predictive mobility is a key factor that can help city officers monitor the performance of traffic [1–3]. Having predictable traffic and travel times is important for traffic management authorities if they are to maintain proper policies and solve traffic problems. Furthermore, to achieve these aims, historical traffic data have become a key factor in the development of prediction models [4]. Accurate travel time prediction assists the various stakeholders such as drivers, travelers, and transportation agencies in transportation, making proper route plans, and managing traffic lights. In addition, predictable traffic facilitates good travel planning and helps reduce environmental problems because the release of carbon dioxide in traffic congestion is higher than that in free-flow traffic [5,6].

City mobility data are essential to the task of precisely predicting travel time on roads [4]. Owing to a collaborative project between King Mongkut's Institute of Technology Ladkrabang and the Department of Land Transport of Thailand, some global positioning system (GPS) tracking data of trucks in Thailand have become available, which were beneficial to our study. Most of the data are for long-distance driving trucks. The dataset entry includes the masked identifier of each vehicle, the timestamp, the latitude, and the longitude, all of which were received from a GPS device for each truck at every minute; some example data are shown in Section 3.1. A thorough literature search revealed that there are research studies that have attempted to conduct travel time prediction, and most of these have taken advantage of time-series data analytics. Several studies, such as those in [7–9], used the Auto-Regressive Integrated Moving Average (ARIMA), a well-known statistical method for predicting travel time. Some machine learning techniques, such as Support Vector Regression (SVR), were also adapted to carry out regression analytics for travel time [10–12]. In addition, in recent years, researchers of travel time prediction have adopted neural network-based approaches such as Long Short-Term Memory (LSTM) [4,13–22] and its enhancement, Convolutional LSTM (ConvLSTM) [14,22], together with attentions [13,14,16] and filters [8,23,24] to improve their prediction models.

To introduce an approach to the prediction of long-distance travel time, we analyzed the time series data according to the average of the travel time every hour from all the trucks between the end points of 10 selected road segments. Our approach uses travel time over a period of 24 h to predict travel time in the following hour. We implemented some techniques from the related work shown in Section 4.1. LSTM provided the best performance against our dataset, so it was selected as a baseline. However, as we interpreted the Mean Absolute Error (MAE) of the LSTM, the error of 27.12 min per 100 km was considered highly inaccurate. Thus, in this study, we aimed to improve the LSTM using techniques that are often adopted in time-series analytics in order to reduce the MAE. In this case, a filter and attention were considered, and it was found that using the self-attention LSTM (SA-LSTM) model together with the Butterworth low-pass filter reduced the MAE to 12.15 min per 100 km.

This manuscript is organized into five sections. First, the background and the objectives of this research are introduced in the Introduction. Second, the technical background from related work is reviewed in the Literature Review. Third, the data preparation, baseline implementation, our approach, and the experiment steps are described in the Materials and Methods section. Next, the results of all the experiments are shown in the Results and Discussion section. Finally, conclusions are drawn, and suggestions are offered in the Conclusion.

2. Literature Review

As noted, when we reviewed the literature, we found various studies on the modeling of travel time prediction. In this section, we describe the background of travel time prediction techniques and review the related studies.

2.1. Travel Time Prediction Techniques

In general, researchers have organized the problem into time-series analytics, then adopted time-series forecasting techniques to create travel time prediction models, such as the ARIMA [7–9], SVR [10–12], LSTM [4,13–22], and ConvLSTM [14,22] models. Some studies proposed improvements using attentions [13,14,16] and filters [8,23,24]. These methods are reviewed and summarized as follows:

Auto-Regressive Integrated Moving Average (ARIMA) is a well-known time-series prediction technique that is commonly used in the domain of finance, especially for data that are seasonal. This technique has been applied for traffic prediction challenges, such as urban roadway travel time prediction from Bluetooth data [7], real-time road traffic state prediction [8], and the prediction of changes in travel time [9]. The ARIMA uses statistical data points to forecast future points. In more detail, the ARIMA has three basic parameters, which are p, d, and q, where p is the number of lag observations (the lag order); d is the number of non-seasonal differences observed from the raw data (the degree of differencing), and q is the moving average window size (the order of the moving average).

Support Vector Regression (SVR) is a regression technique that utilizes the concept of a well-known classification method: the Support Vector Machine (SVM). In our review, we found studies of prediction models using SVR, including travel time of freight transport [10], urban main road travel time [11], and traffic flow prediction [12]. The SVM uses a kernel, which is a mathematical function, to transform data points to aid a hyperplane separation of the data in higher-dimensional space. SVR uses a hyperplane to define decision boundaries to maximize the number of included data points [25]. On the basis of this mechanism, the SVR model can be applied as a time-series forecasting technique. However, an appropriate kernel must be chosen to fit with the characteristics of the data.

Long Short-Term Memory (LSTM) is a type of artificial Recurrent Neural Network (RNN) architecture. Some recent research studies have utilized this method as follows:

- Ran et al. [13] used LSTM and extension to predict travel time in an English road network.
- Daun et al. [4] used LSTM analytics on travel time on freeways using data from inductive loop detectors.

- Wu et al. [14] predicted bus journey times using LSTM.
- Punyo et al. [15] adopted LSTM to estimate bus arrival times and performed an evaluation using MAE.
- Sun and Kim [16] used LSTM with self-attention to predict travel time by grouping the time into hours.
- He et al. [17] created a travel time prediction model using LSTM and trip plans from historical GPS data and road networks.
- Zhang et al. [18] made a travel time prediction using LSTM with multiple lane levels.
- Tran et al. [19] used LSTM for bus travel time prediction by having distance, speed, and start time as the parameters.
- Chen et al. [20] made a travel time prediction model using different methods and found that the LSTM provided the highest performance.
- Agafonov and Yumaganov [21] used LSTM to predict travel time on five routes whose average distances were about 16 km.
- Xu and Rong [22] performed an experiment on Convolutional LSTM (ConvLSTM) and LSTM to predict travel time and found that the LSTM with trip plans had a better performance than that of the ConvLSTM in their dataset.

In more detail, the LSTM uses the connection of the feedback to analyze the sequences of data [26], so it is a well-known technique for time-series prediction. As depicted in Figure 1, LSTM architecture usually contains a sequence of units corresponding to window size. One unit has an input and output, and inside a unit, there is a cell state, a forget gate, an input gate, and an output gate. The cell state (C_t) is a memory from a previous unit (C_{t-1}) that is updated and sent to the next unit, where the gates contain the combination of the current input (x_t) and the previous output (h_{t-1}) activated by either the sigmoid function (*sig*) or the hyperbolic tangent function (*tanh*). In addition, the LSTM can be improved by the addition of attention, such as self-attention, so it becomes a self-attention LSTM (SA-LSTM). The mechanism of an SA-LSTM involves the learning of the association between the current batch of data and the previous parts in a sequence of data. In the case of travel time prediction, this method attempts to choose near-term data points that have a high correlation with 1 [13,14,16].

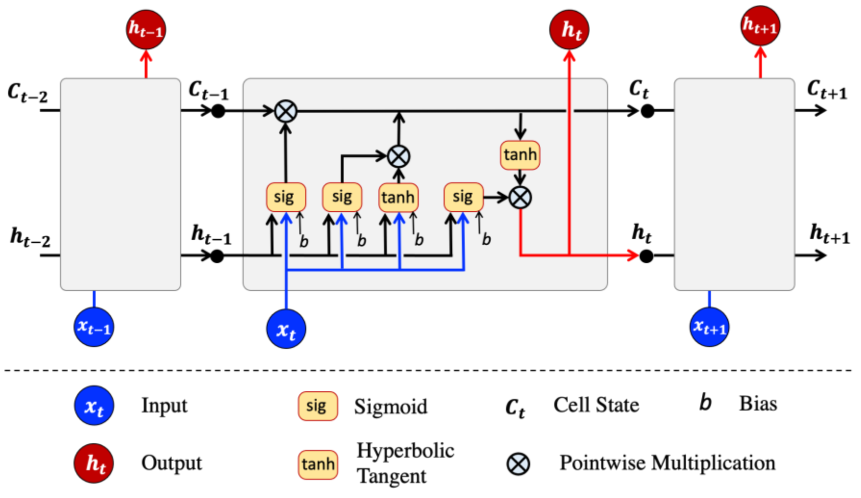


Figure 1. The architecture of a Long Short-Term Memory (LSTM) network.

In addition to the original LSTM, Convolutional Long Short-Term Memory (ConvLSTM) is also used in the travel time prediction problem [14]. The ConvLSTM is an extension of LSTM that uses convolutional layers to process input. This approach is suitable for large input matrices, such as images. There are studies, such as [14], that have taken advantage of this method to predict the travel time of buses and have found that the ConvLSTM prediction model provided the best performance. In addition, the work in [22] also used the ConvLSTM in a travel time prediction experiment.

Moreover, to improve the analytic method, we often required filters to transform the data into a good shape for good-quality inputs for training the model. Thus, methods to clean, denoise, reduce demotion, and transform the dataset were needed. To achieve these goals, some filters were applied, such as a Kalman filter with a neuron network or LSTM [23,24]. As time-series data can be viewed as signal data, some well-known filters, such as the Kalman filter, the Savitzky–Golay filter, and the Butterworth low-pass filter, have been researched [27,28]. First, there is the Kalman filter [21], which has often been used for travel time prediction [23,24]. This filter calculates a joint probability distribution across the variables for each period to smooth the signal data. Second, the Savitzky–Golay filter uses a low-degree polynomial and the linear least-squares method to smooth data points based on adjacent points. Last, the Butterworth low-pass filter is a well-known signal processing filter that operates in a frequency domain. This method uses an algorithm to screen the signal by cutting off the high-frequency signal pass, so it generally helps denoise signal data. Thus, we also attempted to adopt these filter methods and selected one of them to improve the performance of the travel time prediction model.

2.2. Review of Related Studies

Since the results of our work showed that the Self-Attention Long Short-Term Memory (SA-LSTM) model with a Butterworth low-pass filter provides the best performance among all the experiments, this subsection details several research studies that used the LSTM technique and viewed the GPS transactions as time-series data.

First, Ran et al. [13] proposed an LSTM neural network with an attention mechanism for travel time prediction and carried out an experiment from a dataset provided by Highways England. In the raw dataset, the highway had several road segments, and the length of each segment in the dataset was about 18 km. The dataset contained the road segments' names, dates, times, travel times, and distances. Then, the authors of this work sampled the data in intervals of 15 min. The LSTM was tuned to have seven units by which to predict the travel time of the next interval. The experiment found that the MAE of the proposed model was 5.788 s, which was better than the LSTM without attention by about 3.19%.

Second, Chen et al. [20] conducted travel time prediction experiments for Taiwan's provincial highway number 61 with deep learning techniques. The dataset was processed into travel time between two gates of the highway with timestamps. The authors of this work adopted several techniques for long-term prediction, such as LSTM, bidirectional LSTM (BiLSTM), Facebook time-series forecasting model (FBProphet), and an extended ARIMA named the Seasonal Auto-Regressive Integrated Moving Average with eXogenous factors (SARIMAX). In the experiment results, the LSTM, BiLSTM, and SARIMAX showed similar performances. However, in the overall experiment, the LSTM had the lowest Mean Relative Error (MRE), which was a mean absolute error per km.

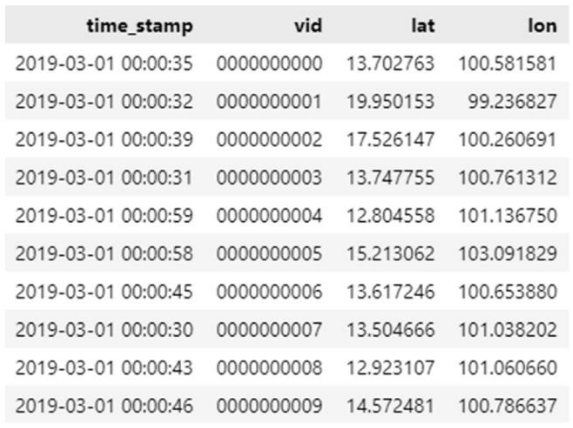
Lastly, Wu et al. [14] proposed a convolutional LSTM model (ConvLSTM) with attention to predicting the travel time of buses. The dataset was a general transit feed specification (GTFS) from Google Transit APIs. The data contained trips, stop sequences, arrival times, and departure times. The authors of this work processed the GTFS data into running times between two stops and waiting times at each stop. The ConvLSTM model predicted both running times and waiting times. The hyperparameter of the model included a learning rate of 0.001, 20 epochs, and a batch size of 16. In addition, the authors allowed the developers to determine an appropriate number of LSTM units according to the datasets and experiments. For the results of the running time prediction, the MAE of the ConvLSTM with attention was 36.328 s, which was about 3.21% higher than that of the ConvLSTM without attention and about 22.95% higher than that of the LSTM without attention.

3. Materials and Methods

In this section, we explain the data preparation of the GPS data of trucks and the implementation of methods from related works for the determination of a baseline and provide a description of our approach and the steps of our experiments.

3.1. Data Preparation

We mainly used GPS data together with road data. The GPS dataset was the GPS tracking data of trucks in Thailand. The dataset collected included the vehicle location and timestamp at every minute from the GPS device, shown in Figure 2. Our work considered data from 2019 because the traffic patterns in that year were not disrupted by the COVID-19 virus. Using the data from trucks, our study focused on the analytics of long-distance roads, and the pre-analysis of these data indicated that many truck GPS data concerned long roads between cities.



time_stamp	vid	lat	lon
2019-03-01 00:00:35	0000000000	13.702763	100.581581
2019-03-01 00:00:32	0000000001	19.950153	99.236827
2019-03-01 00:00:39	0000000002	17.526147	100.260691
2019-03-01 00:00:31	0000000003	13.747755	100.761312
2019-03-01 00:00:59	0000000004	12.804558	101.136750
2019-03-01 00:00:58	0000000005	15.213062	103.091829
2019-03-01 00:00:45	0000000006	13.617246	100.653880
2019-03-01 00:00:30	0000000007	13.504666	101.038202
2019-03-01 00:00:43	0000000008	12.923107	101.060660
2019-03-01 00:00:46	0000000009	14.572481	100.786637

Figure 2. Examples of GPS data of vehicles with data fields that include a date time (timestamp), a masked identifier for each vehicle (vid), a latitude (lat), and a longitude (lon).

In addition, the road data for Thailand were prepared from OpenStreetMap [29]. As there were many roads in the road network data, we chose roads that had a high occurrence of truck GPS data. Highways 1, 2, 4, 7, 9, 32, 35, 41, 304, and 331 were included, and they are shown on the map in Figure 3. Since there were no trucks driving along the entirety of the road lines, some long-distance road segments that had a high number of trucks passing through were selected. For example, we chose 332 km of highway number 1, 232 km of highway number 2, and 219 km of highway number 4, which are shown in Tables 1 and 2 under the columns named Distance (km).

Next, we used the locations of each truck in the GPS dataset to map them to the road segments and selected only trucks that passed through the beginning point and the end point of each road segment. In this case, there were about a hundred million GPS data entries. Since the timestamp of each truck was found at the beginning point and the end point of a road segment, we were able to calculate the travel time of each road segment, as shown in Table 1. In addition, the stopping times, which were calculated from a duration of longer than 20 min in the same location, were excluded. Table 1 shows example data for highway number 1, where each entry is the average travel time from every beginning hour. Thus, this dataset can be viewed as time-series data with an hourly sampling rate. For example, the first data row of Table 1 means that trucks entering the starting point of the road segment highway number 1 during 00:00 and 00:59 took an average travel time of 18,421.14 s (about 5 h and 7 min) to the end point of that road segment. To better understand the dataset, the average hourly travel time of trucks on highway number 1 in 2019 is plotted in Figure 4. In addition, the processed data are summarized in Table 2. There were 81,068 unique trucks with 2,010,270 trips and an average travel time of 203.04 min.

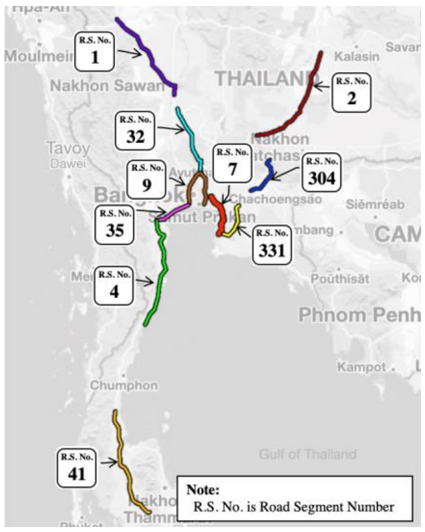


Figure 3. Visualization of polylines of the selected highway sections on a map of Thailand.

Table 1. Average travel time in seconds on a road at every hour of each day in 2019. It is noted that the date format is YYYY-MM-DD where YYYY is a year, MM is a month, and DD is a day.

Road Segment	**Distance (km)**	**Date**	**Hour Duration**	**Travel Time (Seconds)**
Highway No. 1	332	2019-02-14	00:00–00:59	18,421.14
Highway No. 1	332	2019-02-14	01:00–01:59	16,412.50
Highway No. 1	332	2019-02-14	02:00–02:59	15,809.55
Highway No. 1	332	2019-02-14	03:00–03:59	15,206.60
Highway No. 1	332	2019-02-14	04:00–04:59	16,552.00
Highway No. 1	332	2019-02-14	05:00–05:59	17,110.75
Highway No. 1	332	2019-02-14	06:00–06:59	14,216.83
Highway No. 1	332	2019-02-14	07:00–07:59	15,873.13
Highway No. 1	332	2019-02-14	08:00–08:59	15,494.25
Highway No. 1	332	2019-02-14	09:00–09:59	20,464.50
Highway No. 1	332	2019-02-14	10:00–10:59	14,278.20
Highway No. 1	332	2019-02-14	11:00–11:59	16,780.00
Highway No. 1	332	2019-02-14	12:00–12:59	14,343.60
...	...	...	...	...

Table 2. Summary of the data of trips passing through the 10 selected road segments.

Road Segment	**Distance (km)**	**Number of Trucks**	**Number of Trips**	**Average Travel Time (mins)**
Highway No. 1	332	9079	71,416	259.96
Highway No. 2	232	8147	53,087	336.68
Highway No. 4	219	15,837	141,157	287.76
Highway No. 7	95	14,018	297,478	220.28
Highway No. 9	149	48,109	755,804	94.78
Highway No. 32	143	10,368	92,304	171.67
Highway No. 35	75	25,827	317,787	96.47
Highway No. 41	242	8676	72,072	293.36
Highway No. 304	104	9741	95,855	138.94
Highway No. 331	80	11,245	113,310	130.48
Summary	**167.1** (Average)	**81,068** (Unique Count)	**2,010,270** (Total)	**203.04** (Average)

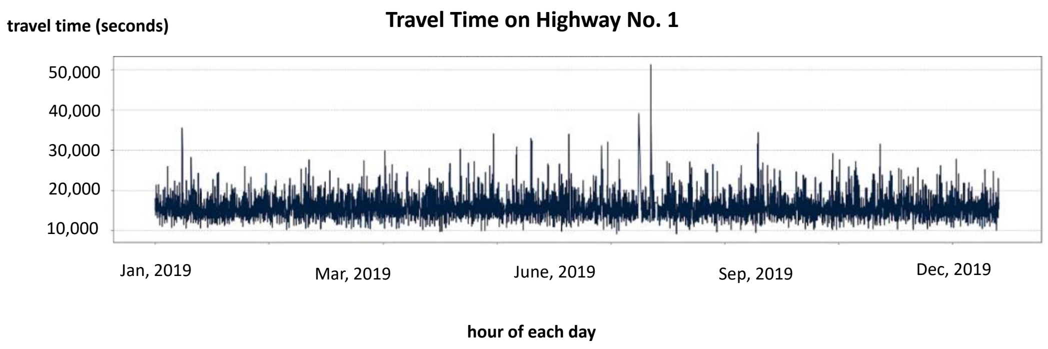


Figure 4. Line chart of the average hourly travel times from Table 1, where the x-axis shows intervals of hours in chronological order, and the y-axis is average travel times of each hour in seconds.

3.2. Baseline Implementation

Based on the review of the time-series analytic techniques discussed in previous related work, we considered ARIMA [7], SVR [25], LSTM [26], and ConvLSTM [14] for the preliminary analysis. The details of these techniques are described in the following subsections. After this, we conducted an experiment to select which of these techniques would provide the best performance as a baseline in our study. The pre-processed dataset that we adopted and examples of it are shown in Table 1, and the steps of our experiment are described in Section 3.4.

3.3. Our Approach

In the first experiment explained in Section 3.4, the LSTM showed the best performance. The MAE was 27.12 min per 100 km, as shown in Section 4.1. This meant that we could use the original LSTM to predict travel time with an error of about 27.12 min for every 100 km. Thus, the LSTM was selected as a baseline, and our challenge was to improve the LSTM, increasing the performance by minimizing the MAE. After obtaining the raw data, we used a filter to process the data in order for it to be an input of the LSTM with attention, as shown in Figure 5.

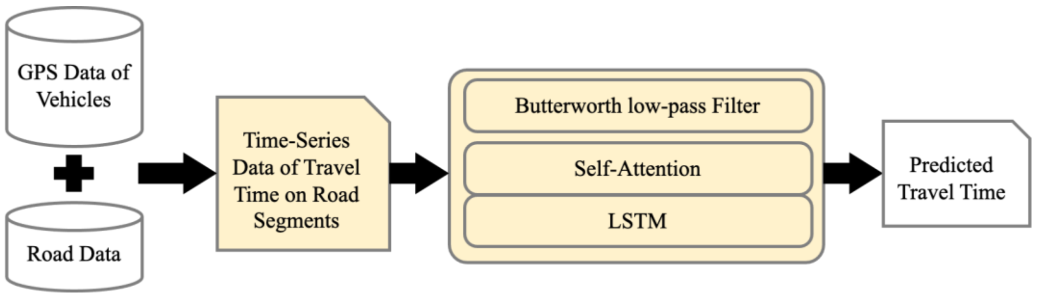

GPS Data of Vehicles
Road Data
Time-Series Data of Travel Time on Road Segments
Butterworth low-pass Filter
Self-Attention
LSTM
Predicted Travel Time

Figure 5. Our overall approach. The time-series data, which are created from Global Positioning System (GPS) data and road data, were passed through a filter to be used as an input of the Self-Attention Long Short-Term Memory (LSTM). The predicted travel time was the output.

3.3.1. Time-Series Data of Travel Time on Road Segments

To create the time-series data of travel time on road segments as input data, the GPS vehicle and road data were processed as follows.

First, the GPS vehicle data were the set of relations expressed by

$$(\text{vid, ts, lat, lon}) \in \text{GPSTX}, \tag{1}$$

where GPSTX is a set of the GPS transactions of the vehicles. The elements were the vehicle identifier (vid), timestamp (ts), latitude (lat), and longitude (lon). The format of the timestamp was "YYYY-MM-DD hh:mm:ss", where YYYY, MM, DD, hh, mm, and ss indicate the year, month, day, hour, minute, and second, respectively. A typical example of data corresponding to this expression is displayed in Figure 1.

Second, the road data including the polyline of coordinates of each road were formulated as

$$(\text{rid}, \text{RPL}) \in \text{ROADS and} \tag{2}$$

$$(\text{lat}, \text{lon}) \in \text{RPL}, \tag{3}$$

where ROADS is a set of road shapes, which are relations including a road identifier (rid) and a road polyline (RPL). The RPL is an ordered set of pairs of latitude (lat) and longitude (lon) points that form the line of a road. An example visualization of some roads in Thailand is presented in Figure 3.

Next, the locations (lat and lon) of each member of GPSTX were mapped to the polyline of each road, so any vehicles passing through every point on each road were documented. After this, the road segments that had high occurrences of vehicles passing through them were selected.

Finally, the timestamps (ts) of the vehicles between the endpoints of each road segment were calculated and became travel time entries. Then, these values were averaged according to hours and presented as time-series data of the travel time on each road segment, which is described by the following expression:

$$(\text{rid}, \text{dh}, \text{tt}) \in \text{TTR}, \tag{4}$$

where TTR is a set of travel time on road segments. The equation elements are road id (rid), date–hour (dh), and average travel time (tt) of a road in that hour. An example of the data is shown in Table 1, where the rid is presented by the road name instead of the road identifier. It should be noted that long-duration parked vehicles were eliminated. This was calculated by finding the duration that a vehicle had stayed in the same latitude and longitude and that exceeded a pre-defined threshold, such as parking for longer than 20 min. The results of this step, which were a time-series dataset of travel time on road segments, became the input for the analytics in the next modules.

3.3.2. Filtering Method

In this section, we used a filtering method to pre-process the time-series data TTR. There are several filtering methods, such as the Kalman, Savitzky–Golay, and Butterworth low-pass methods. In order to find a suitable filtering method, we conducted experiments as described in Section 3.4.3. Based on our experimental results in Section 4.3, we chose the Butterworth low-pass filter [28] as part of our model.

3.3.3. Self-Attention LSTM (SA-LSTM)

Our main approach was to use the SA-LSTM to predict the travel time. We learned the TTR data, which were processed by a filter method, to create one model for one road, so we had 10 models for 10 selected road segments. In addition, we used a window size of 24 due to the observation of hourly patterns across different days. However, the window size can be configured according to the pattern of the data. In addition, the decision to use self-attention or operate without attention depends on the experiment. In our case, the comparison between the LSTM only and the SA-LSTM models guided us to choose the SA-LSTM.

Finally, our approach is formulated as expressed in the following equation:

$$\text{tt}_{\text{rid}}^{\hat{\text{dh}}} = (\text{SA\_LSTM} \circ \text{BF})\left(\text{TTR}_{\text{rid}}^{[\text{dh}_{-24}:\text{dh}_{-1}]}\right), \tag{5}$$

where $\text{tt}_{\text{rid}}^{\hat{\text{dh}}}$ denotes a predicted travel time (tt) of a given road segment (rid) at a given date–hour (dh). The formula is the composite of the SA_LSTM (SA_LSTM) model and the

Butterworth low-pass filter (BF), where the input $TTR_{rid}^{[dh_{-24}:dh_{-1}]}$ is a set of travel time (TTR) of a given road segment (rid) between 24 h ($[dh_{-24} : dh_{-1}]$) before the given dh.

3.3.4. Architecture of Our Travel Time Prediction Model

This subsection describes the architecture of our travel time prediction model. The model is demonstrated in Figure 6, after all the experiments were completed and with all the adjusted parameters. Our model has six layers: input data, a Butterworth low-pass filter, a self-attention layer, an LSTM layer, a fully connected layer, and an output. It was assumed that the model had already been trained from historical data. Thus, some variables, such as the weight vector of query (W_Q), the weight vector of key (W_K), the weight vector of value (W_V), the hidden state input vector (H_0), the cell state input vector (C_0), the weight tensor of the LSTM (W_{LSTM}), and the weight matrix (W_{FC}) of the artificial neural network in the fully connected layer, were already assigned.

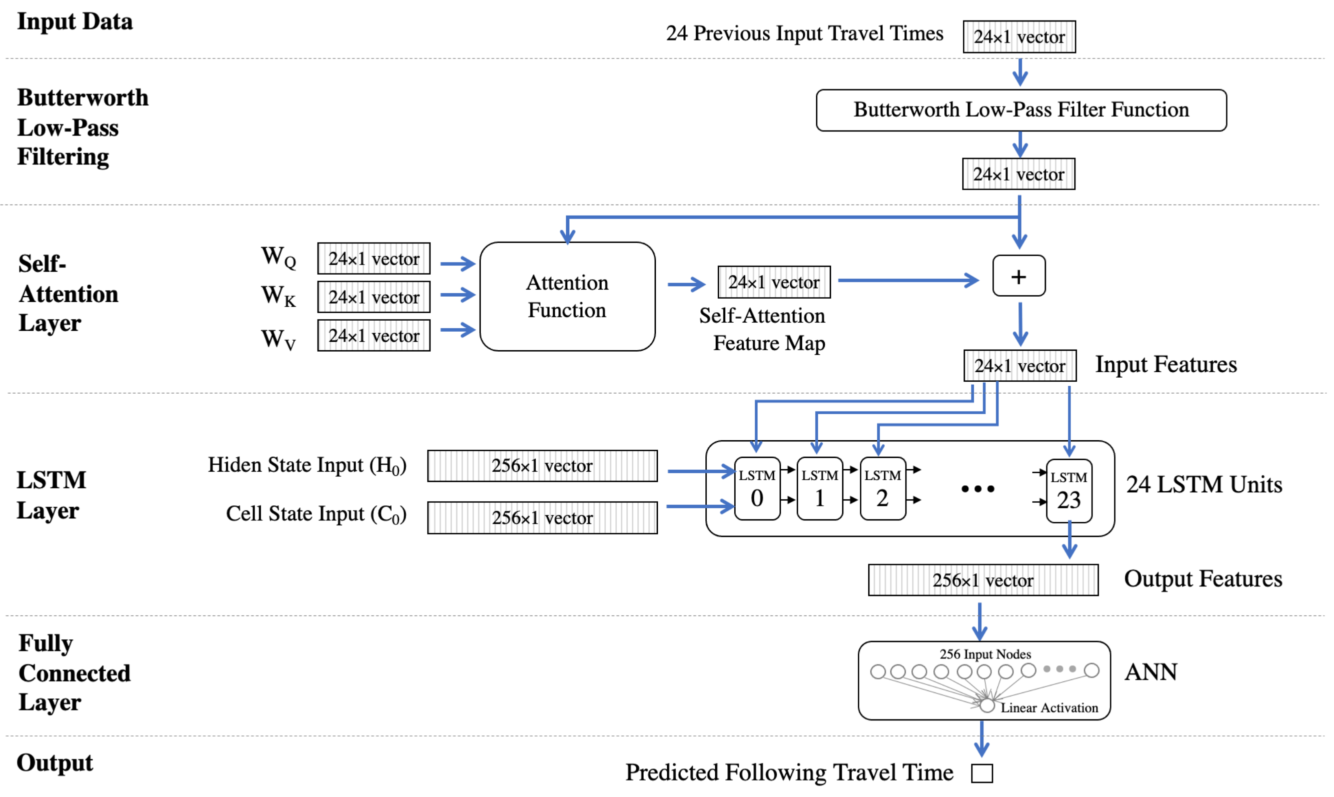


Figure 6. The architecture of our travel time prediction model.

Input Data

The input data were the sequences of previous travel times over a period of 24 h that were selected from TTR (Equation (4)) with a given road segment. This is a 24-dimension (24-dim) vector. In the case of predicting road segment A at 07:00 on one day, the average hourly travel time along road segment A between 07:00 and 24:00 of the previous day and between 00:00 and 06:00 of the current day needs to be collected.

Butterworth Low-Pass Filter

The task of this layer is to process the input vector by the Butterworth low-pass filtering. This resulted in the 24-dim vector.

Self-Attention Layer

The self-attention layer has an attention function that multiples the input travel time vector with the trained weight vectors: W_Q, W_K, and W_V, becoming vectors Q, K, and V, respectively. Then, it calculates the vectors Q, K, and V by the $\mathrm{SoftMax}\left(\frac{Q\cdot K^T}{\sqrt{24}}\right)\cdot V$ equation, where the SoftMax is a function expressed by $\mathrm{SoftMax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{J} e^{z_j}}$ for $i = 1, \ldots, J$. This resulted in the 24-dim vector of the self-attention feature map. Then, the input vector from the previous layer is added by the self-attention feature map vector, so it becomes a 24-dim input feature vector.

LSTM Layer

The LSTM layer consists of 24 LSTM units. All of the LSTM units are constructed by the trained weight tensor (W_{LSTM}). In addition, the trained 256-dim vectors of the hidden state input vector (H_0) and the cell state input vector (C_0) are inputs of the first LSTM unit. Then, the result of the last LSTM unit is used as an output feature, which is a 256-dim vector.

Fully Connected Layer

The fully connected layer is an artificial neural network (ANN). This ANN is constructed by the trained weight matrix (W_{FC}). In this case, there are 256 input nodes whose values are from the 256-dim output feature vector. Then, the ANN generates one output node with a linear activation function.

Output

The output from the fully connected layer is the predicted travel time of the following hour along a given road segment.

In addition, we wrote the following pseudocode to demonstrate our model in detail, corresponding to Figure 6. The input was a 24-dim input vector of the travel time (input_ttr) and the trained variables; the output was a predicted travel time of the following hour. The trained variables were selected according to the road segment from input_ttr. This means that it needed to prepare trained variables for each road segment. In the pseudocode, BF() is a function of the Butterworth low-pass filter; SoftMax() is a SoftMax function; LSTM() was used to construct an LSTM model based on an input cell state, an input hidden state, and a weight tensor; and ANN() was used to construct an ANN model based on the input weight matrix with the linear activation function (Algorithm 1).

Algorithm 1. Model: Travel Time Prediction

```
Input:   input_ttr          //a vector representing the input of previous travel times (24-dim)
         $W_Q$, $W_K$, $W_V$      //3 vectors for self-attention (24-dim)
         $C_0$                 //a cell state input vector (256-dim)
         $H_0$                 //a hidden state input vector (256-dim)
         $W_{LSTM}$              //a weight tensor of LSTM for 24 units
         $W_{FC}$                //a weight matrix for the fully connected layer (256-dim)
Output: pred_tt             //the prediction of a following travel time (scalar)
1: //Butterworth Low-Pass Filter
2: input_ttr $\leftarrow$ BF(input_ttr)                                   //24-dim vector
3: //Self-Attention Layer
4: Q $\leftarrow$ $W_Q \times$ input_ttr                                   //24-dim vector
5: K $\leftarrow$ $W_K \times$ input_ttr                                   //24-dim vector
6: V $\leftarrow$ $W_V \times$ input_ttr                                   //24-dim vector
7: sa_map $\leftarrow$ SoftMax($Q\cdot K^T \div \sqrt{24}$) $\cdot V$                      //24-dim vector
8: input_features $\leftarrow$ input_ttr + sa_map                          //24-dim vector
9: //LSTM Layer
10: lstm_model $\leftarrow$ LSTM($C_0$, $H_0$, $W_{LSTM}$)                       //24 LSTM units
11: output_features $\leftarrow$ lstm_model.predict(input_features)          //256-dim vector
12: //Fully Connected Layer
```

Algorithm 1. *Cont.*

```
13: ann_model ← ANN(W_FC, 'linear')                      //256 input nodes and 1 output node
14: pred_tt ← ann_model.predict(output_features)         //scalar
15: RETURN pred_tt
```

3.4. Experiments

As mentioned, we aimed to use a time-series technique to make a prediction of travel time along each selected road segment. We conducted three experiments to model our method. First, the time-series-based prediction techniques ARIMA, SVR, LSTM, and ConvLSTM were compared to decide which one would be the baseline. Second, self-attention was used to improve the LSTM by comparison with the original LSTM. Finally, several filter methods used to pre-process the time-series data were compared.

In our experiment, we evaluated the performance of each model using a train–test split. In this case, the ordered TTR dataset of each road was split into third sets: a training set (60%), a validation set (20%), and a test set (20%), as shown in Figure 7a. The training set was used to train the model; the validation set was used to validate the loss to the adjusted hyperparameters during learning; the test set was used to test the performance of the trained model. In every iteration of the predictions, there were 24 continuous input data points as a sliding window size denoting 24 h, and the model used these to produce a predicted output representing the next hour. In other words, we used a travel time of 24 h to predict the following-hour travel time iteratively, as depicted in Figure 7b. To evaluate the model, we used the Mean Absolute Error (MAE) [14], as shown in Equation (6), where N is the number of test data; i is the test sample index; $y^{(i)}$ is the actual travel time of the test sample at index i; and $y^{\hat{(i)}}$ is the predicted travel time of the test sample at index i. Our evaluations were reported by normalizing the MAE by 100 km. For example, if the model of road number 1, whose distance is 332 km, provides an MAE of 15, it is calculated by $15 \times 100 \div 332$, giving 4.52.

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|y^{(i)} - y^{\hat{(i)}}\right| \tag{6}$$

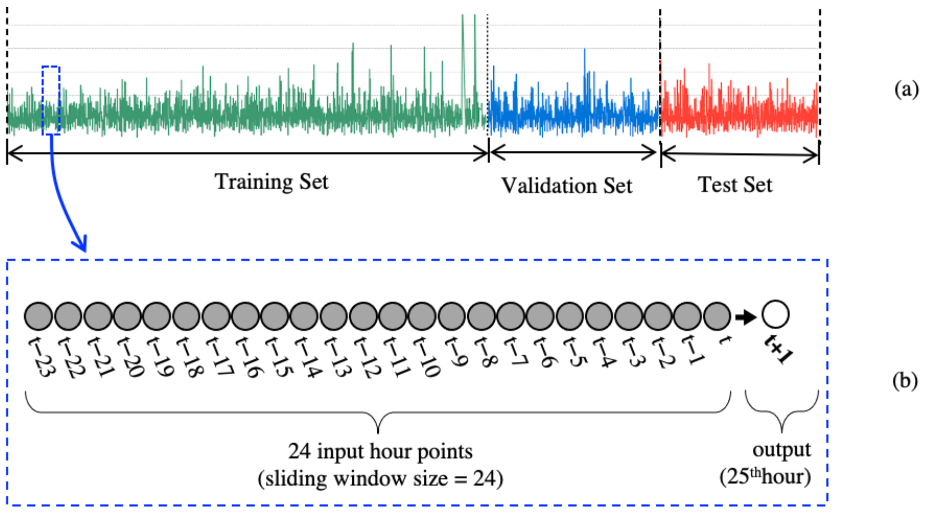


Figure 7. (**a**) The split of the time-series data into a training set and a test set, and (**b**) an example of an input with a sliding window size of 24 and one output.

3.4.1. Experiment: Choosing a Suitable Baseline

This experiment was conducted to compare four techniques from related works to determine our baseline. These techniques were ARIMA [7], SVR [25], LSTM [26], and ConvLSTM [14]. The tuned hyperparameters of these techniques were as follows:

- ARIMA: p, q, and d were based on observations of the data of each road, such as the settings of the highway number 1 being $p = 7$, $q = 0$, and $d = 12$.
- SVR: sliding window size = 24, kernel = RBF (Radial Basis Function), C (for L2 Regularization) = 1, epsilon = 0.1, gamma = 0.1.
- LSTM: input of LSTM units = 24, number of LSTM layers = 1, hidden LSTM size = 256, and dropout = 0.25.
- ConvLSTM: The setting of the ConvLSTM was reused from the LSTM. Additionally, the settings of the convolutional layer were 3 convolutional layers with size = 32, kernel size = 2, and max pooling size = 2.

The results of this experiment are shown in Section 4.1. It shows that the LSTM provides the best performance among the models from the related studies, so the LSTM model was chosen as the baseline. In the subsequent experiments, improvements to and comparisons against the LSTM were performed.

3.4.2. Experiment: Choosing a Self-Attention

As mentioned, self-attention is a mechanism that can be adopted to improve the LSTM in regard to the time-series problem [14]. In this step, we conducted an experiment to compare the original LSTM from the previous experiment and the LSTM with self-attention, called the SA-LSTM. The tuned hyperparameters of both techniques are described in the following list.

- LSTM: The setting was the same as that in the experiment in Section 3.4.1.
- SA-LSTM: The setting was the same as that of the LSTM by adding input size = 1 and output size = 24 (for self-attention).

3.4.3. Experiment: Selecting a Suitable Filter Method

As discussed, filter methods can be used to improve the performance of the prediction of time-series data by processing the input data before building a prediction model; thus, in this experiment, we aimed to find a suitable filter method for our work. Since the results of the SA-LSTM were better than those of the original LSTM, as presented in Section 4.2, this experiment focused on using filter techniques with the SA-LSTM. Thus, in this experiment, the SA-LSTM without a filter technique was compared to the SA-LSTM with the selected filter techniques, which were the Kalman filter, the Savitzky–Golay filter, and the Butterworth low-pass filter.

- SA-LSTM: The setting was the same as that in the experiment in Section 3.4.2.
- Kalman: diagonal matrix = 0.1, measurement = 2
- Savitzky–Golay: windows size = 9, polynomial order = 3
- Butterworth low-pass: cut off = 3, frequency = 10, order = 2

4. Results and Discussions

The results of the three experiments are shown and discussed in this section. Since the distances between road segments were different, for example, the length of the selected road segment of highway number 1 was 332 km, and that of highway number 331 was 80 km, as mentioned in Section 4.4, the error of each model was normalized by the length, so the results were reported as MAE per 100 km instead of the original MAE. Thus, an MAE value such as 31.25 for the ARIMA in Section 4.1 can be interpreted as follows: the evaluation of the ARIMA technique with the test set provided an error of $\pm$31.25 min within 100 km.

4.1. Results of Baselines

Firstly, the results of the experiment for the selection of a baseline described in Section 3.4.1 are shown in Table 3. In this table, the MAE of the LSTM technique was 27.12, which was better than the result for ARIMA, SVR, and ConvLSTM. Thus, the LSTM model was selected as the baseline.

Table 3. Performance report of the time-series prediction techniques: Autoregressive Integrated Moving Average (ARIMA), Support Vector Regression (SVR), Long Short-Term Memory (LSTM), and Convolutional LSTM (ConvLSTM), evaluated by Mean Absolute Error (MAE).

Techniques	MAE (minutes) per 100 km
ARIMA	31.25
SVR	29.96
LSTM	**27.12**
ConvLSTM	28.32

4.2. Results of Self-Attention LSTM

Secondly, Table 4 shows the results of the experiment in Section 3.4.2, performed in order to choose a self-attention for the LSTM. The table compares the LSTM only and the LSTM with self-attention (SA-LSTM). Based on the experiment, the MAE of the SA-LSTM was 26.89, which was better than that of the LSTM only.

Table 4. Performance report of comparison of the LSTM and SA-LSTM.

Techniques	MAE (minutes) per 100 km
LSTM (only)	27.12
SA-LSTM (Self-Attention LSTM)	**26.89**

4.3. Results of Filters

Lastly, the results of the experiment for the selection of an appropriate filter method, as explained in Section 3.4.3, are shown in Table 5. According to our test, the MAE of the SA-LSTM with the Butterworth low-pass filter was 12.15, and this was the best result among all the compared methods. Thus, the Butterworth low-pass filter was selected as part of our approach.

Table 5. Performance report of the comparison of SA-LSTM and SA-LSTM with a filter technique.

Techniques	MAE (minutes) per 100 km
SA-LSTM	26.89
SA-LSTM + Kalman filter	20.95
SA-LSTM + Savitzky–Golay filter	20.96
SA-LSTM + Butterworth low-pass filter	**12.15**

4.4. Results of Our Approach against the Baseline

Moreover, as part of a closer look at our experiments, we analyzed the results of each road segment, and these are summarized in Table 6. In this table, the name of each road segment and its length in km, the MAE per 100 km between the baseline and our approach, and the percent improvement are shown. In this case, the baseline was the LSTM only, while our approach used the SA-LSTM with the Butterworth low-pass filter method. According to the table, the average MAE of our approach for all the road segments was 12.15, while the baseline was 27.12.

Table 6. A comparison of the baseline and our approach for all road segments.

Road Segment	**Distance (km)**	**MAE (minutes) per 100 km**		**% Improvement**
		Baseline	**Our Approach**	
Highway No. 1	332	9.19	4.04	56.08%
Highway No. 2	232	22.06	9.15	58.53%
Highway No. 4	219	13.70	6.01	56.15%
Highway No. 7	95	49.66	25.41	48.83%
Highway No. 9	149	29.89	12.88	56.92%
Highway No. 32	143	31.65	14.29	54.86%
Highway No. 35	75	17.35	8.23	52.57%
Highway No. 41	242	14.16	6.36	55.11%
Highway No. 304	104	19.47	8.16	58.07%
Highway No. 331	80	64.03	26.98	57.87%
Average		**27.12**	**12.15**	**55.20%**
Standard Deviation (STD)		**17.47**	**8.01**	**0.03**

4.5. Computation Time

This was an additional experiment that observed the computation time of each method from the previous experiments. This experiment was performed on a computer with a four-core central processor, 8 gigabyte of memory, and a Linux operating system. The results are shown in Table 7. For the deep learning approaches from the ConvLSTM onwards, it was evaluated by 50 epochs. The SA-LSTM with Butterworth low-pass filter had a computation time similar to that of the other LSTM-based approaches.

Table 7. Computation times of each technique in our experiment.

Techniques	**Computation Time (Seconds)**	
	Training	**Predicting**
ARIMA	146.56	0.04
SVR	8.16	0.55
ConvLSTM	493.86	1.25
LSTM (only)	337.08	0.58
SA-LSTM	320.14	0.58
SA-LSTM + Kalman filter	327.13	0.58
SA-LSTM + Savitzky–Golay filter	331.26	0.58
SA-LSTM + Butterworth low-pass filter	314.61	0.58

4.6. Discussion

The aim of our work was to improve the accuracy of a travel time prediction model using a time-series technique. Our dataset was the GPS transactions of trucks in Thailand, as depicted in Figure 3. A closer look at the data indicated that the location of trucks with a timestamp could be viewed as time-series data. Thus, we sampled the raw data according to the average hourly travel times of 10 road segments of a long distance, as presented in Table 1. To create an approach for the generation of a travel time prediction model, several time-series forecasting techniques were studied from related works. (1) In the first experiment, we attempted to find a suitable time-series technique for our baseline. The result of the experiment was that the LSTM provided the best performance. The results from Table 3 were interpreted as follows: the LSTM model provided an error of ± 27.12 min in every 100 km. This error range, being almost half an hour, was regarded as being too high in error. Thus, in this case, this issue became the main focus of our study, and it was our challenge to solve this problem. (2) In the second experiment, the LSTM was improved with the addition of self-attention, becoming an SA-LSTM. The results presented in Table 4 show that the SA-LSTM was better than the LSTM. Although the MAE was slightly reduced, this was a sign of enhancement of the model. (3) In the last experiment, we tried to find an appropriate filter method for data cleansing. The results in Table 5, using the Butterworth low-pass filter with SA-LSTM, were the best among all the experiments. This may have been due to the noise of the original data having a high impact on the performance of the prediction model. When this filter was applied, the MAE of the SA-LSTM was reduced from 26.89 to 12.15.

Based on the experimental results, our best approach was to use the SA-LSTM model with the Butterworth low-pass filter. As we analyzed for each road, as shown in Table 6, we can conclude that the improvement of most road segments was greater than 50%, and the total improvement was 55.20%. Although the MAEs of all the road segments had high variance, as indicated by the standard deviation (STD) in the table, the percent improvements of all the road segments were not much different. This means that our method achieved a relatively stable improvement rate. This result can also be visualized in the line chart in Figure 8. It shows values of actual travel time points (solid black line) and the predicted values (red dashed line) of a few days from all selected road segments.

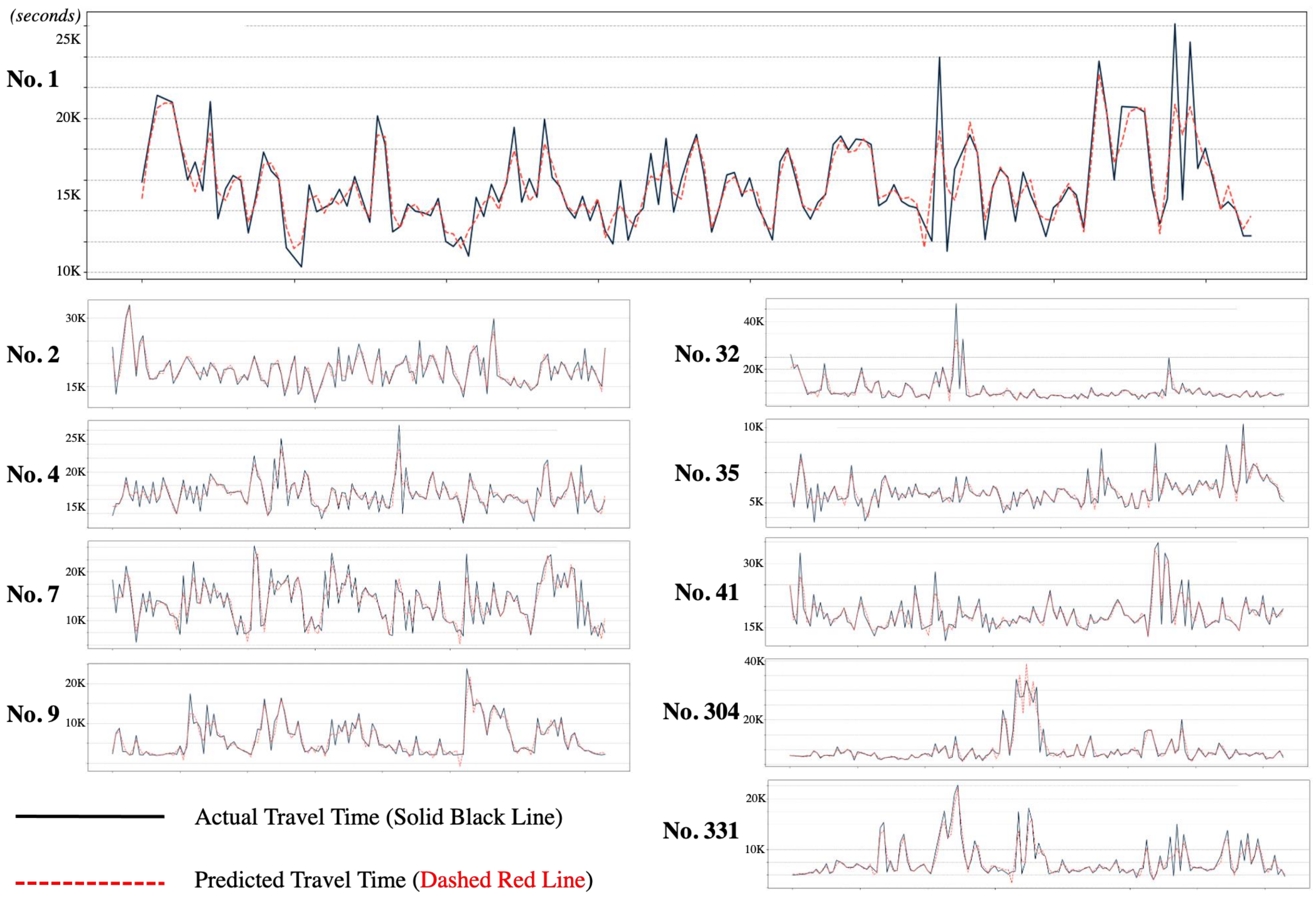


Figure 8. Line chart visualizing the comparison between actual values (black solid line) and predicted values (red dashed line) using the SA-LSTM model with Butterworth low-pass filter from each road segment of a few days.

In addition, our approach specifically emphasized the GPS data and viewed the GPS transactions as time-series data. The road data were also used to support the selection of road segments that had a high occurrence of vehicles passing through them. Thus, to work with our approach, we recommend that machine-learning developers collect GPS transactions of vehicles, including vehicle identifiers, latitude, longitude, and timestamps. It should be noted that the road data were further used to find reference points to identify road sections. Then, both datasets were analyzed into time series of travel time of each selected road segment, and this dataset was used as the input for our method.

As reviewed in the related studies in Section 2, several works, including the LSTM model discussed by Ran et al. [13], the LSTM model of Chen et al. [20], and the ConvLSTM model of Wu et al. [14], inspired our work. First, the model of [13] is closest to our work in terms of an LSTM with attention. It used interval data of 15 min as data points and used seven data points as an input of the LSTM and forwarded these to the attention layer to predict the travel time of the following data point. For our approach, we used the LSTM with self-attention and considered 24 input units at one-hour intervals. The experiments from [13] and our work also show that using the LSTM with attention is better than that without attention; the experiment results from [13] showed a 3.19% improvement, while our work had an improvement of 0.84%. Although this improvement is somewhat low, using attention can be a good means of increasing the performance of the LSTM model. Second, the model of [20] considered an LSTM without attention. It also used interval data as the input data in order to predict the subsequent data points. For the long-term prediction, the goal was to use 182 days to predict the travel time of the following 30 days. In our work, we attempted to use the travel time of the previous 24 h to predict the travel time of the following hour. Finally, the ConvLSTM model from [14] used the convolutional layers after LSTM. Although the ConvLSTM did not give a good result in our experiment, it evidenced that the use of self-attention can slightly improve the model. This work also stated that the LSTM played a key role when near-term data were highly correlated to the subsequent travel time. In addition to the selected studies [13,14,20], our model considered the selection of a filter method, which was the Butterworth low-pass filter, to process the input data. This filter method is generally applied in time-series analytics [23,24]; however, it might not be necessary for the three selected studies. As we learned from the related studies and our approach, the architecture of the travel time prediction model can be customized. In the case of using the LSTM as a core technique, the hyperparameters, such as the number of LSTM units, must be adjusted. In addition, other assembly techniques, such as attention layers, convolutional layers, and filter methods, should be considered. This consideration is based on an experimental environment, including datasets, goals, and evaluation methods. According to our dataset and the results of the experiments, our travel time prediction model became a Self-Attention LSTM with a Butterworth low-pass filter that can process the travel time of the previous 24 h to estimate that of the following hour.

Since this study was strongly focused on the GPS data, the historical travel time data were a key factor in the prediction of future travel time. The advantage of our study is that it attempted to use only time-series of GPS data without other features to predict the travel time of a subsequent hour. This results in one prediction model per road segment. In addition, some other studies [16,18,20,22] considered other features, such as weather, lane level, and the day of the week. In this case, our method framework can be enhanced by the inclusion of other aspects such as weather, road repair schedules, and local festivals that need much more study. Thus, further study should consider other factors so as to improve the travel time prediction model.

The potential application of our work is to provide a service for the travel time prediction of commercial trucks. In this case, the government department that owns the big GPS data should calculate the travel time of every route in every following hour, then provide the predicted data to consumers. The consumers might be logistic firms that need precisely estimated travel times for a logistics cycle to improve their logistics plans and monitor unusual situations during delivery processes. To the best of our knowledge, a function for predicting the travel time of commercial trucks is not presented in any online map application. In this case, it can add more value to both data owners and data consumers.

To this end, as mentioned in the first section, the monitoring of some anomalies in road traffic can help city policymakers and officers achieve solutions to traffic problems more quickly and effectively. When road traffic must be predicted under usual traffic conditions, any irregular state that is highly different from the predicted one can be observed [1–3]. As our work shows, historical traffic data can be used to make travel time prediction models. In the future, if we can create values from data in any perspective of the traffic domain, such as speed and flow prediction, route prediction, and accident prediction, then smart mobility will become even more realistic. In any case, our work is another significant step in the development of data-driven smart cities in Thailand and also in other countries.

5. Conclusions

The aim of this study was to introduce an approach for the prediction of travel time for vehicles on long-distance road segments in Thailand. The input data that we collected were truck GPS transactions and included vehicle identifiers, timestamps, latitudes, and longitudes. The raw data, together with road data from Thailand, were processed into time-series data of the average hourly travel time on each road segment in 2019, and the time-series data were split into a training set and a test set, where the sliding window size was 24. In our study, the LSTM was selected as a baseline and had an MAE of 27.12 min per 100 km. After studying the data in detail and conducting several experiments, the SA-LSTM with the Butterworth low-pass filter was adopted as our method, and this provided an error of ± 12.15 min for every 100 km, which was 55.20% better than that of the baseline. The results of our experiments suggest that having a predictive travel time using GPS data for each road is practical and is a necessary factor in the improvement of smart mobility. Thus, our work can contribute to the development of a data-driven smart city.

However, our work is an early step in the area of travel time prediction in Thailand. In the future, other factors such as weather data and road repair schedules might be considered. In addition, a future plan, which will require more effort and a bigger budget, is to collect the GPS data of other types of vehicles such as personal cars and, indeed, the whole road network in Thailand. Further research can then enrich travel time prediction models.

Author Contributions: Conceptualization, R.C. and A.S.; methodology, R.C.; software, N.A., S.I. and N.B.; validation, R.C., N.A., S.I., N.B. and A.S.; formal analysis, R.C, N.A, S.I. and N.B.; investigation, R.C.; resources, R.C. and A.S.; writing—original draft preparation, R.C.; writing—review and editing, R.C.; visualization, R.C., N.A., S.I. and N.B.; supervision, R.C. and A.S.; project administration, R.C.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Bibri, S.E.; Krogstie, J. The emerging data–driven Smart City and its innovative applied solutions for sustainability: The cases of London and Barcelona. *Energy Inform.* **2020**, *3*, 5. [CrossRef]
2. Nagy, A.M.; Simon, V. Survey on traffic prediction in smart cities. *Pervasive Mob. Comput.* **2018**, *50*, 148–163. [CrossRef]
3. Lin, Y.; Li, R. Real-time traffic accidents post-impact prediction: Based on crowdsourcing data. *Accid. Anal. Prev.* **2020**, *145*, 105696. [CrossRef] [PubMed]
4. Duan, Y.; Yisheng, L.V.; Wang, F.-Y. Travel time prediction with LSTM neural network. In Proceedings of the 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), Rio de Janeiro, Brazil, 1–4 November 2016; pp. 1053–1058.
5. Oh, S.; Byon, Y.-J.; Jang, K.; Yeo, H. Short-term travel-time prediction on highway: A review on model-based approach. *KSCE J. Civ. Eng.* **2017**, *22*, 298–310. [CrossRef]
6. Qiu, B.; Fan, W. Machine Learning Based Short-Term Travel Time Prediction: Numerical Results and Comparative Analyses. *Sustainability* **2021**, *13*, 7454. [CrossRef]
7. Carrese, S.; Cipriani, E.; Crisalli, U.; Gemma, A.; Mannini, L. Bluetooth Traffic Data for Urban Travel Time Forecast. *Transp. Res. Procedia* **2021**, *52*, 236–243. [CrossRef]
8. Xu, D.-W.; Wang, Y.-D.; Jia, L.-M.; Qin, Y.; Dong, H.-H. Real-time road traffic state prediction based on ARIMA and Kalman filter. *Front. Inf. Technol. Electron. Eng.* **2017**, *18*, 287–302. [CrossRef]

9. Reza, R.Z.; Pulugurtha, S.S. Forecasting short-term relative changes in travel time on a freeway. *Case Stud. Transp. Policy* **2019**, *7*, 205–217. [CrossRef]
10. Servos, N.; Liu, X.; Teucke, M.; Freitag, M. Travel Time Prediction in a Multimodal Freight Transport Relation Using Machine Learning Algorithms. *Logistics* **2019**, *4*, 1. [CrossRef]
11. Philip, A.M.; Ramadurai, G.; Vanajakshi, L. Urban Arterial Travel Time Prediction Using Support Vector Regression. *Transp. Dev. Econ.* **2018**, *4*, 7. [CrossRef]
12. Li, C.; Xu, P. Application on traffic flow prediction of machine learning in intelligent transportation. *Neural Comput. Appl.* **2020**, *33*, 613–624. [CrossRef]
13. Ran, X.; Shan, Z.; Fang, Y.; Lin, C. An LSTM-Based Method with Attention Mechanism for Travel Time Prediction. *Sensors* **2019**, *19*, 861. [CrossRef] [PubMed]
14. Wu, J.; Wu, Q.; Shen, J.; Cai, C. Towards Attention-Based Convolutional Long Short-Term Memory for Travel Time Prediction of Bus Journeys. *Sensors* **2020**, *20*, 3354. [CrossRef] [PubMed]
15. Panyo, K.; Bootkrajang, J.; Inkeaw, P.; Chaijaruwanich, J. Bus Arrival Time Estimation for Public Transportation System Using LSTM. In Proceedings of the 2020—5th International Conference on Information Technology (InCIT), Chonburi, Thailand, 21–22 October 2020; pp. 134–138. [CrossRef]
16. Sun, J.; Kim, J. Joint prediction of next location and travel time from urban vehicle trajectories using long short-term memory neural networks. *Transp. Res. Part C Emerg. Technol.* **2021**, *128*, 103114. [CrossRef]
17. He, P.; Jiang, G.; Lam, S.-K.; Sun, Y. Learning heterogeneous traffic patterns for travel time prediction of bus journeys. *Inf. Sci.* **2019**, *512*, 1394–1406. [CrossRef]
18. Zhang, S.; Guan, D.; Liu, H. Research on Lane-level Travel Time Prediction Method Based on Gated Recurrent Neural Network. *J. Phys. Conf. Ser.* **2020**, *1693*. [CrossRef]
19. Tran, L.; Mun, M.Y.; Lim, M.; Yamato, J.; Huh, N.; Shahabi, C. DeepTRANS: A deep learning system for public bus travel time estimation using traffic forecasting. *Proc. VLDB Endow.* **2020**, *13*, 2957–2960. [CrossRef]
20. Chen, M.-Y.; Chiang, H.-S.; Yang, K.-J. Constructing Cooperative Intelligent Transport Systems for Travel Time Prediction with Deep Learning Approaches. *IEEE Trans. Intell. Transp. Syst.* **2022**, 1–10. [CrossRef]
21. Agafonov, A.; Yumaganov, A. Bus Arrival Time Prediction with LSTM Neural Network. In Proceedings of the International Symposium on Neural Networks, Moscow, Russia, 10–12 July 2019; pp. 11–18. [CrossRef]
22. Xu, W.; Rong, W. Neural network model based on travel planning for travel time prediction. *J. Phys. Conf. Ser.* **2021**, *1883*, 012010. [CrossRef]
23. Aljamal, M.A.; Abdelghaffar, H.M.; Rakha, H.A. Developing a Neural–Kalman Filtering Approach for Estimating Traffic Stream Density Using Probe Vehicle Data. *Sensors* **2019**, *19*, 4325. [CrossRef]
24. Fang, W.; Cai, W.; Fan, B.; Yan, J.; Zhou, T. Kalman-LSTM Model for Short-term Traffic Flow Forecasting. In Proceedings of the 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 12–14 March 2021; Volume 5, pp. 1604–1608. [CrossRef]
25. Zhang, F.; O'Donnell, L.J. Support vector regression. In *Machine Learning*; Elsevier: Amsterdam, The Netherlands, 2020; pp. 123–140.
26. Van Houdt, G.; Mosquera, C.; Nápoles, G. A review on the long short-term memory model. *Artif. Intell. Rev.* **2020**, *53*, 5929–5955. [CrossRef]
27. Arsene, C. Design of Deep Convolutional Neural Network Architectures for Denoising Electrocardiographic Signals. In Proceedings of the 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), Via del Mar, Chile, 27–29 October 2020; pp. 1–8.
28. Van Van Breugel, F.; Kutz, J.N.; Brunton, B.W. Numerical Differentiation of Noisy Data: A Unifying Multi-Objective Optimization Framework. *IEEE Access* **2020**, *8*, 196865–196877. [CrossRef] [PubMed]
29. OpenStreetMap Data for This Region: Thailand. OpenStreetMap. 2018. Available online: https://download.geofabrik.de/asia/thailand.html (accessed on 9 October 2021).