Travel time prediction for an intelligent transportation system based on a data-driven feature selection method considering temporal correlation

Kandiri, Amirreza; Ghiasi, Ramin; Nogal, Maria; Teixeira, Rui

DOI
10.1016/j.treng.2024.100272

Publication date
2024

Document Version
Final published version

Published in
Transportation Engineering

Transportation Engineering 18 (2024) 100272

Contents lists available at ScienceDirect

Transportation Engineering

journal homepage: www.sciencedirect.com/journal/transportation-engineering

Travel time prediction for an intelligent transportation system based on a data-driven feature selection method considering temporal correlation

Amirreza Kandiri [a,*], Ramin Ghiasi [a], Maria Nogal [b], Rui Teixeira [a]

[a] School of Civil Engineering University College Dublin, Ireland
[b] Faculty of Civil Engineering & Geosciences, Delft University of Technology, Netherlands

ARTICLE INFO

Keywords:
Feature Selection
Data Reduction, Travel Time Prediction, Traffic Management, Data-driven predictive models
Intelligent Transportation System

ABSTRACT

Travel-time prediction is a critical component of Intelligent Transportation Systems (ITS), offering vital information for tasks such as accident detection, congestion management, and traffic flow optimisation. Accurate predictions are highly dependent on the selection of relevant features. In this study, a two-stage methodology is proposed which consists of two layers of Optimisation Algorithm and one Data-Driven method (OA2DD) to enhance the accuracy and efficiency of travel-time prediction. The first stage involves an offline process where interconnected optimisation algorithms are employed to identify the optimal set of features and determine the most effective machine learning model architecture. In the second stage, the real-time process utilises the optimised model to predict travel times using new data from previously unseen parts of the dataset. The proposed OA2DD method was applied to a case study on the M50 motorway in Dublin. Results show that OA2DD improves the convergence curve and reduces the number of selected features by up to 50 %, leading to a 56 % reduction in computational costs. Furthermore, using the selected features from OA2DD, reduced the prediction error by up to 29 % compared to the full feature set and other feature selection methods, demonstrating the method's effectiveness and robustness.

1. Introduction

Globally, traffic congestion has become a pressing issue with widespread consequences, such as higher accident rates, elevated air pollution, increased fuel consumption, less predictable travel times, and a decline in societal health [1]. Given these challenges, it becomes crucial to implement an effective Intelligent Transportation System (ITS) that can dynamically assess the traffic situation [2]. The functioning of the traffic system is intricately linked to the handling of fundamental traffic data [3]. To do so, predicting different aspects of traffic is one of the most important steps [4].

Numerous studies have been studying the prediction of different aspects of traffic such as traffic flow from regional [5-7] and network perspectives [8-10], speed [11,12], occupancy [13,14], travel time from path perspective that considers the travel time as the duration that vehicles spend in specific route(s) from entering to exiting [15,16] and origin-destination perspective that focuses on vehicles' total trip travel time [17,18]. Some other studies have focused on predicting the travel demand [19,20]. The importance of predicting travel time is not limited to road networks. Recent studies show that approaches on prediction and usage are captivating interest in other modes of transport, such as the railway systems [21,22].

Unlike flow, occupancy, or speed, travel time is more readily understandable to humans. The interpretation of flow, occupancy, or speed can vary depending on the type of road being analysed [23]. The prediction of travel time is a crucial element of ITS [24], and it has a significant role in the deployment of Advanced Traveler Information Systems (ATIS) and Advanced Traffic Management Systems (ATMS) [25]. The information regarding travel time serves as valuable input or auxiliary data for accident detection, congestion control, dynamic navigation, and other related applications. Consequently, studying travel-time prediction holds significant importance [25].

Travel time prediction methods have been categorised from different points of view in the literature. However, one of the most common approaches is to divide them into a) model-based and b) data-driven methods [25]. Model-based methods use traffic parameters such as flow and speed to build a traffic model and predict future travel time. Two common model-based methods are the cell transmission model [26,27] and queuing theory [28]. However, data-driven methods are the mainstream methods today [3], which uses datasets extracted from the

* Corresponding author.
E-mail address: amirreza.kandiri@ucdconnect.ie (A. Kandiri).

https://doi.org/10.1016/j.treng.2024.100272
Received 3 July 2024; Received in revised form 4 September 2024; Accepted 5 September 2024
Available online 7 September 2024

past and finds the potential in them to predict future travel time. Machine Learning (ML) techniques such as Artificial Neural networks (ANN) [29-31], Support Vector Regression (SVR) [32-34], and Deep Learning [35,36] are the commonly used techniques in data-driven methods.

Abdi and Amrit [37] in a review paper on prediction travel time and arrival time, gathered information on previous studies including input features. However, unity in the features is not observed among the studies. For instance, Kumar et al [38] used the day of the week, time of the day (TOD), and jam density, while Chiabaut and Faitout [39] selected the TOD, length of the section, speed, and congestion. In another study, Wang et al [40] only used speed, distance, and departure time. Finding the correct features to use as input is crucially important. If the data-driven methods are not given enough information (i.e., features), they fail to predict accurately [41]. On the other hand, having irrelevant and redundant features can reduce the accuracy alongside increasing the computational cost [42]. Therefore, having a systematic approach to choose the necessary features is the first step in predicting travel time accurately. Consequently, this paper introduces a holistic methodology, from data collection to prediction, with a novel feature selection method consisting of two Optimisation Algorithms (OAs) and a Data-Driven predictive model (DDPM) named OA2DD.

The OA2DD method is based on the selection of features using the wrapper method, where feature subsets are selected via OAs and assessed using predictive algorithms. To demonstrate the efficacy of the OA2DD method, the Salp Swarm Algorithm (SSA) is utilized as an optimiser, with ANN employed for prediction. Furthermore, the optimal choice of the predictive algorithm structure has a significant effect on its accuracy. Therefore, the structure of the ANN is also optimised using another OA, i.e. SSA in this case. As a result, the proposed methodology yields more accurate travel time prediction requiring less computational resources. It is noted that in this work, travel time is studied from the path perspective, however, OA2DD logic can be implemented on the origin-destination perspective as well.

The method is applied to a case study involving the M50 motorway in Dublin, Ireland. The M50 motorway holds significant importance for both the local and national transportation infrastructure as the busiest road in Ireland [43] and numerous studies have been conducted on the M50, examining various aspects of this motorway. Corbally et al [44] predicted the duration of incidents in M50 using ML algorithms. Rogers and Darcy [45] studied the impact of toll booths on traffic flow. Kemp and O'Mahony [46] identified the latency in travel time prediction in M50, quantified its effects and proposed a model to remove it.

Overall, the main contributions of the current study are as follows:

- The study introduces a two-stage methodology for travel time prediction, consisting of a novel feature selection method called OA2DD and a layer of DDPM.
- The OA2DD is a novel feature selection method based on the wrapper method. Unlike a conventional wrapper method that selects and investigates the impact of each feature on the output individually, OA2DD chooses the features according to the temporal correlation among them, where feature subsets are selected via OAs and assessed using predictive algorithms.

The current study is structured as follows: Section 2 provides the theoretical background of the ML and OA used in this study and also feature selection concept. Section 3 comprehensively explains the Methodology from data collection to real-time prediction of travel time and introduces the novel OA2DD feature selection method, Section 4 illustrates the Methodology through a case study employing SUMO simulation, detailing each step along with the presentation and discussion of results. Lastly, Section 6 outlines the primary conclusions drawn and suggests potential avenues for further exploration.

2. Theoretical background

2.1. Salp Swarm Algorithm

Salp Swarm Algorithm is a metaheuristic OA widely employed in various engineering fields due to its simplicity and adaptability to diverse problems [47]. Inspired by the swarm intelligence of salps, SSA mimics the efficient coordination and movement of these marine organisms in search of a food source. The salp population in SSA is divided into leaders and followers, and their positions are defined in a n-dimensional search space, where n represents the number of decision variables in the optimisation problem. The following equations present how their positions are updated in each iteration [48].

$$P_i^1 = \left\{ \begin{array}{l} FS_i + r_1((UB_i - LB_i)r_2 + LB_i)\ r_3 \geq 0 \\ FS_i - r_1((UB_i - LB_i)r_2 + LB_i)\ r_3 < 0 \end{array} \right\} \tag{1}$$

$$r_1 = 2e^{-\left(\frac{4h}{H}\right)^2} \tag{2}$$

$$UP_i^j = \frac{1}{2}\left(P_i^j + P_i^{j-1}\right) \tag{3}$$

where, P_i^1, P_i^j, and UP_i^j, are the *i-th* dimension of leader, *j-th* follower and updated *j-th* follower's position, respectively. Moreover, H, h, FS_i, UB_i, and LB_i, and are the maximum number of iterations, the current iteration, the *i-th* dimension of the food source, upper, and lower bounds, respectively. Finally, r_2 and r_3 are random numbers between 0 and 1. Fig. 1, shows the algorithm's pseudo code.

2.2. Artificial Neural Network

Artificial Neural Network is an ML technique inspired by the human neural system and its remarkable learning abilities. Essentially, ANN enables the learning of relationships between input values and an output quantity, even in the case of highly complex associations. The widely used ANN training technique is feed-forward back-propagation (FFBP) [49,50]. FFBP-based ANNs typically feature at least one hidden layer situated between an input layer and an output layer. Nodes in these layers are connected to every node in the subsequent layer through weight values.

In the feedforward phase, information moves from the input to the output layer to generate predictions based on the input data and the current weights. During back-propagation, weights are adjusted using training algorithms like Levenberg-Marquardt, Bayesian Regularization, or Scaled Conjugate Gradient to minimise prediction errors [49]. Levenberg-Marquardt is an extension of the Quasi-Newton method and avoids the need to compute the Hessian matrix for solving non-linear least squares problems. The weights and biases (wb_{k+1}) are updated according to the following formula [51]:

$$wb_{k+1} = wb_k - \left(V^K V + \mu Z\right)^{-1} V^K e$$

where V is the Jacobian matrix containing the first derivatives of the network errors with respect to the weights and biases, e is the error vector, Z is the identity matrix, and μ is a damping factor. Neurons in the hidden layers receive inputs that are weighted based on their relevance from the previous layer. These inputs are then summed, with a bias added to the total, and the result is passed through an activation function. This process introduces non-linearities, enabling the network to model complex patterns. In contrast, output neurons usually don't use an activation function, and input layer neurons simply pass the data into the network without applying weights or biases. Training an ANN involves adjusting both weights and biases to optimise performance. Weights determine how much influence a neuron's output has on the next layer, while biases act as thresholds that allow neurons to activate

```
Start
    Generate the initial population of salps randomly with respect to lower and upper bounds
    while t < T
        Calculate each salp's fitness
        Set the best salp as the food source
        Determine r1 using Eq. (2)
        for j = 1 : Salp population size
            for i = 1 : Number of dimensions
            Generate r2 and r3 randomly between 0 and 1
                if j == 1
                    Update ith dimension of the leader using Eq. (1) with respect to bounds
                else
                    Update ith dimension of the jth salp using Eq. (3) with respect to bounds
                endif
            endfor
        endfor
    endwhile
    Return the food source
Finish
```

Fig. 1. SSA's pseudo code.

even when all weighted inputs are zero. This adds flexibility, enabling the network to learn patterns more effectively and improve prediction accuracy. Biases, as constant values, help the network adapt to different patterns. Fig. 2 provides a schematic of an ANN and neurons in its hidden and output layers.

2.3. Feature selection

One of the primary objectives in data modelling and classification is to make predictions based on the training data and available features. Dealing with vast datasets characterized by a high-dimensional feature space poses significant challenges in ML tasks [52]. When there are numerous irrelevant and redundant features within the initial feature set, the utilization of dimensional reduction becomes crucial [53]. Dimensional reduction is an essential technique for eliminating these undesirable features. This process not only enhances the performance of ML algorithms but also reduces computational cost by eliminating irrelevant and redundant features [54-56].

Reducing the problem dimensions by identifying the important features not only reduces computational load and increases the accuracy, but also helps understanding the underlying relationship between the key features and travel time. Furthermore, exclusively collecting essential features can result in saving time and resources in data collection [57]. Hence, after extracting features from the collected raw data, a feature selection process must be done. Feature selection methods typically explore the solution space to minimise the redundancy of the selected features and maximise their relevance to the target class [58]. These methods can be based on a) filter models and b)

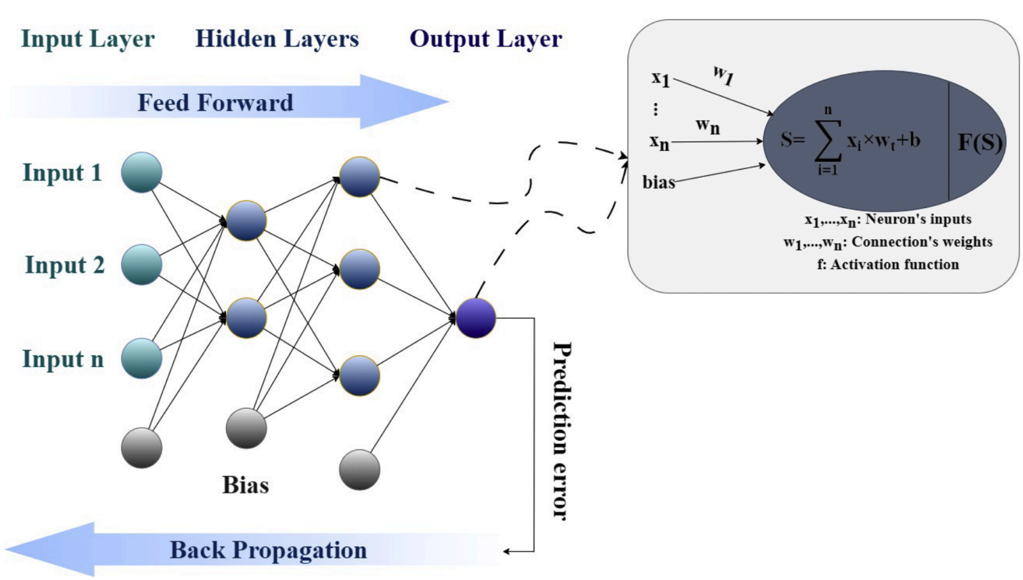

Fig. 2. Schematic ANN and neuron.

wrapper models.

The filter model assesses the significance of features without employing any learning algorithm, making methods based on this model generally rapid. Within these models, features undergo evaluation and prioritization using information-theoretic metrics, and then those with the highest ranks are selected [59]. Wrapper-based methods integrate a supervised learning algorithm into the feature selection process, selecting features through the subset evaluation technique. The selection of features takes into account correlations and dependencies between them, with consideration for the bias of the prediction algorithm to optimise overall performance [60]. Wrapper-based methods, in general, utilize iterative search processes where each iteration of the learning model guides the population of solutions toward the best solution [56]. Wrapper models have a higher computational cost compared to filter models due to their use of learning algorithms. Nevertheless, an intelligent optimisation algorithm-based feature selection method can explore the relationship between features making no assumptions [42]. Moreover, they can choose a subset of features with more compatibility with DDPM since they use a similar technique inside. However, there is limited use for them to select features for predicting traffic states and even less for predicting travel time [42].

2.4. Explored features in travel time prediction

There is a notable variety in the features used by studies in travel time predictions. In recent years the importance of used features for predicting travel time and their impacts on the accuracy of the predictions has gotten more attention. Although there is no unity in the used features in the literature, there are some studies that studied these features and improved the travel time accuracy by extracting effective features from their case studies. However, overall, there is a lack of focus on feature selection. Most studies focused on increasing prediction accuracy by extracting new features and adding these features to the input vector rather than being exploratory in identifying the features or combinations that improve accuracy in the most efficient form.

For instance, Tang et al. [61] developed a tensor-based context-aware approach to predict travel time. To capture the contextual information of traffic conditions, this approach extracts historical, geographical and spatial-temporal features. In another study, Abdollahi et al. [36] used feature learning, feature extraction, and clustering algorithms in their developed predictor to increase the accuracy of the prediction. The used dataset in this study was the New York City Taxi and Limousine Commission. Therefore, they used features such as passenger count, fare, and tip. Moreover, Kawatani et al. [62] studied predicting bus travel time between two stops. They realized that adding the travel time of the previous bus between the same two stops and the travel time of the same bus between previous stops could increase the accuracy of the prediction. Shen et al. [63] proposed a neural network for travel time prediction based on tensor decomposition and graph embedding, to address the insufficient feature extraction of extract road network structure and travel speed.

In one of the few studies on feature selection for predicting travel time, Jia et al. [64] used a feature selection method named sample entropy. This feature selection method is used for time series prediction problems. It calculates the complexity of each subsequence to find the one that increases the accuracy of the prediction in methods such as long short-term memory. In another study that was conducted on predicting travel time using urban big data, Zou et al. [65] studied the impact of various features on predicting results by sequentially adding these features into the model. Although this can be considered as a form of feature selection, this approach can be computationally costly if not impossible when the number of features increases. Moreover, Li et al. [66] extracted 14 candidate features from GPS and weather stations. Then they used XGBoost and light GBM to select the 10 most effective features. Although this study used a systematic approach for feature selection, it does not study spatial or temporal correlation between features. XGBoost was also used in another study on predicting travel time by Kang et al. [67]. In this study, spatio-temporal correlation between features was studied.

3. Methodology

In this work, the proposed Methodology consists of two parts. The first part is an offline process, whose main outcome is a list of features constituting a trained DDPM to predict the travel time. The second part is an online process which includes training the DDPM using real-time data to yield a real-time prediction of the travel time. Fig. 3 depicts the Methodology and its components. The rest of this section explains the different steps of the offline process.

3.1. Feature extraction

Three types of input including volume, time of the day, week, month, or year, and speed have been used to predict travel time [68]. However, this information needs to be extracted from the raw data collected from detectors. A common way to collect traffic data is by using sensors placed in fixed locations along the road. These sensors detect vehicles that pass by. Some typical sensors used for this purpose include inductive-loop detectors, cameras, and radar [1]. To extract the needed information, in the first step, a day should be divided into time intervals. Then, in the case of volume, upstream and downstream detectors are used to count the number of vehicles within the understudy segment of the highway. Moreover, traffic has a daily, weekly, monthly, seasonal, and annual cycle. Therefore, the time of the day, day of the week, week of the month, and month of the year should be considered. Finally, the average speed of each lane (if accessible), or the speed limit of each lane if they have different speed limits should be used (see Fig. 4).

Moreover, there is a temporal correlation in the time series of travel time. In other words, there is a correlation between the mean travel time of the current time interval and the same value of the previous time intervals [69]. Therefore, the same features- i.e. volume, time of the day, week, month, or year, and speed- of the previous time intervals should be considered as well. However, extracted features from the previous time intervals are different from the extracted features from the current time interval. For the previous time intervals, information about the time of the day, week, month, or year, and speed is not needed since the model already has the information from the current time interval. However, depending on how congested the traffic is, how long the segment is, and how long the time interval is, some of the vehicles from previous time intervals may have not exited the segment yet. Therefore, the travel time of the vehicles that exited the segment during the previous time intervals should be considered instead.

Although travel time is an important piece of information and draws a sensible image of the traffic, more information is needed to be able to illustrate a completely understandable image. For instance, two time intervals can have the same mean travel time when one of them has slow consistent traffic, while vehicles can speed up first and get stuck in congested traffic later in the other time interval. Therefore, another parameter named waiting time should be used to address that issue. Waiting time is the accumulative time that vehicles spend in the segment while driving slower than a given threshold which is 0.36 km/h in this work. The waiting time of the vehicles that exited the segment during the previous time intervals is considered a candidate feature as well. Then all the extracted features will be put inside the candidate feature pool.

3.2. Candidate features pool

Candidate features pool is the set of all the features that potentially increase the accuracy of the predicted travel time without (or with the least) increase in the redundancy of the selected features. The extracted features from the current time interval are the TOD (i.e., week, month,

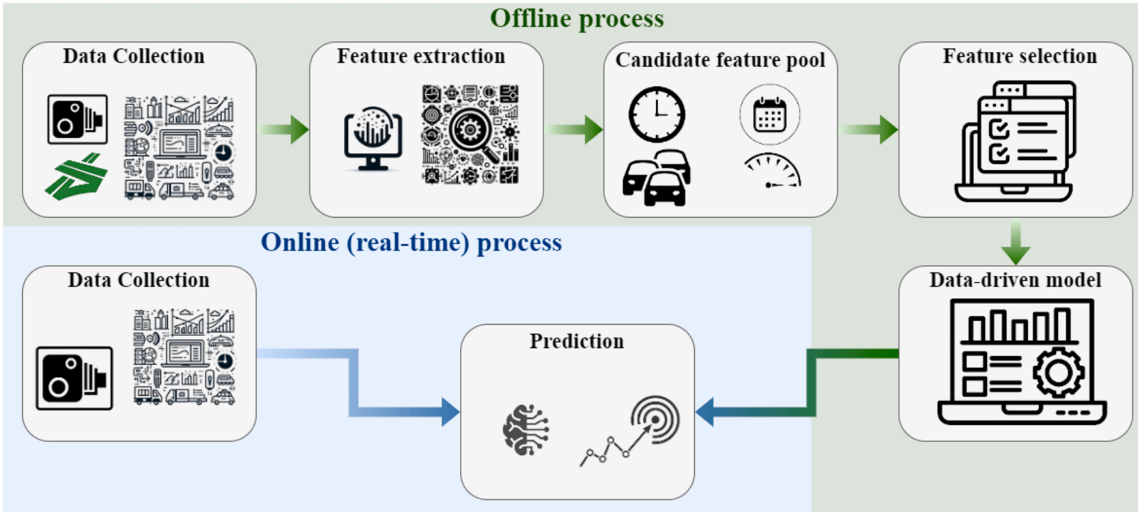

Fig. 3. Methodology's different parts and the interconnections.

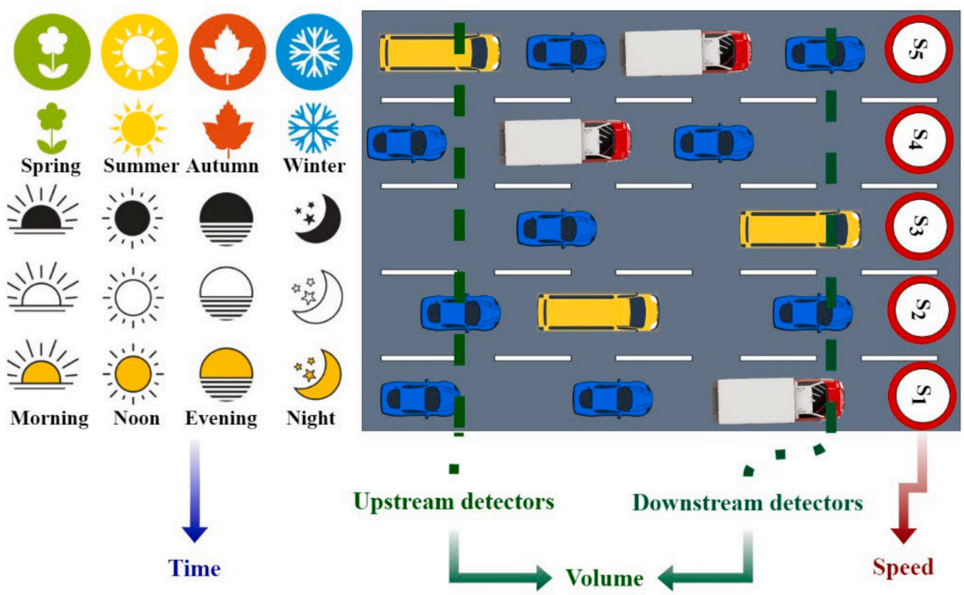

Fig. 4. Aspects to be considered in the feature extraction.

and year if accessible), speed of each lane (i.e., speed limit or average speed if accessible), the number of vehicles entering the segment during the time interval, and the number of vehicles that are already inside the segment when the time interval starts. Moreover, the extracted features from the selected number of previous time intervals are the number of vehicles entering the segment during those time intervals, the number of vehicles that already are in the segment when those intervals started, the mean travel time (MT) and the mean waiting time (MW) of the vehicles that exit the segment during those intervals. Selecting the number of previous time intervals depends on the accessible data and computational budget. Table 1 summarises the candidate features.

Table 1
Candidate features.

Current time interval (t)	t-1	t-2	…	t-n
Time of the day	Entering vehicles	Entering vehicles		Entering vehicles
Per-lane speeds	Existing vehicles	Existing vehicles		Existing vehicles
Existing vehicles	Mean travel time	Mean travel time		Mean travel time
	Mean waiting time	Mean waiting time		Mean waiting time

3.3. Feature selection

Feature extraction can encompass a broad spectrum of candidate features. Nevertheless, not all extracted features contribute to enhancing forecasting accuracy, as there might be irrelevant or redundant features within the extracted set [69]. The elimination of these features has the potential to further improve forecasting accuracy or, at the very least, maintain the current level of accuracy while reducing the complexity of the predicting model [70-72]. Existing feature selections either rank the potential features based on their importance (filter models) or decide whether a feature should be selected or not. These methods consider features individually. However, in the current study, the majority of the features are sequential and extracted due to their temporal correlation;

hence, some of them cannot be selected without the others. For instance, the travel time of the *t-5th* time interval cannot be selected without the travel time of the next time intervals up until the current one. To address this limitation, a novel feature selection method hybridizing two OAs and a DDPM named OA2DD is proposed in this study. In the current study, to be able to explain the Method, an ANN as the DDPM and a SSA as an optimiser are used, which are explained in Section 2.

3.3.1. OA2DD

As mentioned before, the main concept of OA2DD is based on the wrapper method in which an OA explores different subsets of features and imports them to a DDPM as its input. Then, based on the accuracy of the predicted output, the wrapper model tries to find the subset of features that maximises the accuracy. However, most of the DDPMs have adjustment parameters that need to be set before the start. In the case of ANN, the number of hidden layers and the number of neurons in each

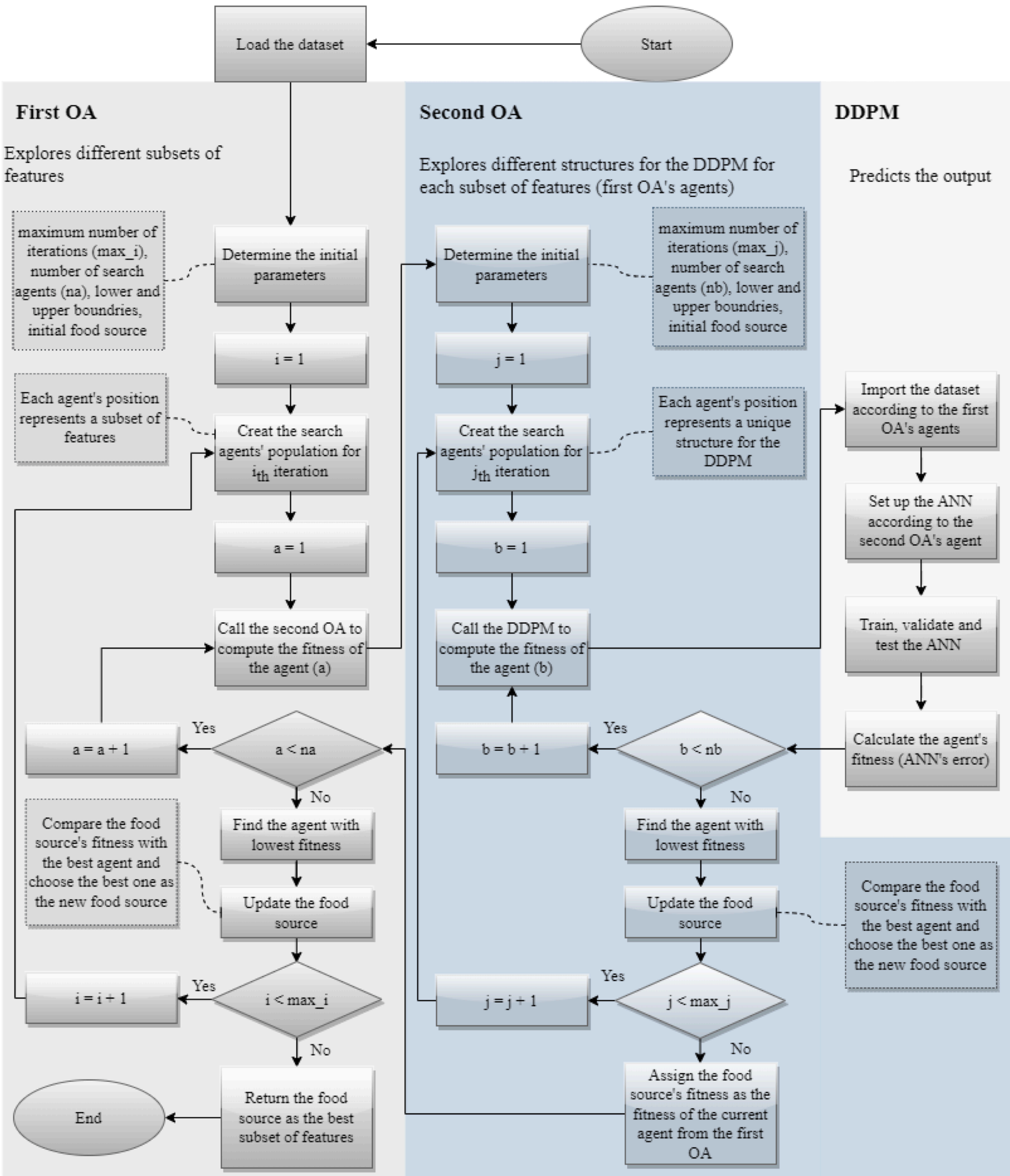

Fig. 5. OA2DD flowchart.

layer need to be set. These numbers that build the ANN's structure crucially impact the ANN's performance and accuracy. Therefore, the structure of an ANN needs to be optimised according to a given feature subset to reach the maximum accuracy possible for that subset of features.

The OA2DD consists of three different parts. First, there is an OA that explores different subsets of features, i.e., the first OA. Then, there is another OA inside the first one, which explores different adjustment parameters for the DDPM for each feature subset, i.e., the second OA. It should be noted that the two OAs work as bi-level optimisation, whereas the second OA is fully nested in the first one. Finally, a DDPM inside the second OA takes the selected features by the first OA as inputs, adapts to the given adjustment parameters chosen by the second OA, and predicts the output after being trained. In other words, the DDPM's error serves as the second OA's objective function, and the hybridization of these two serves as the first OA's objective function. This way, the minimum error related to each feature subset can be acquired; hence, a fair comparison between feature subsets can be conducted and the best subset with the overall minimum error can be chosen. Fig. 5 illustrates OA2DD's process, and the rest of this section explains OAs and DDPM individually. a) The first OA

The first OA's job is to explore different subsets of features to select the subset that causes the minimum error in predicting the output, i.e., travel time. In the first step, it determines the initial parameters. Then, it randomly creates the search agents' population with respect to the lower and upper bounds. Each search agent's position represents a subset of features. The extracted features in the candidate features pool are divided into five categories and the selected features from each category by a search agent are represented by a cell in its position (see Fig. 6). The features in the first category are the ones related to the current time interval, hence they are always being selected since they have important impacts on the output. In each of the other categories, there is a temporal correlation among the members. Therefore, none of them can be chosen for a subset unless all the members before them are chosen. For instance, existing vehicles at the start of *t*-4 cannot be chosen unless the same values of *t*-3, *t*-2, and *t*-1 are chosen. As a result, the agent only carries a number in its position for each category which represents how many members of that category from the top are selected. For example, an agent with the position of [3,3,1,0,1] represents a subset of time, of the day, week, month, or year, speeds, and existing vehicles at the start of *t*, entering vehicles during *t*-1, entering vehicles during *t*-2, entering vehicles during *t*-3, existing vehicles at the start of *t*-2 and waiting time of *t*-1. The number of considered previous time intervals, i.e., *n*, depends on the problem properties and can be set by the user. After generating every agent's position, it calls the second OA for each agent to calculate their fitness. Then, the agent with the best fitness, i.e., the lowest error, is compared with the food source and if it has a better fitness, it replaces the food source. Finally, search agents' positions are updated for the next iteration and the same process is conducted until the maximum number of iterations is reached. The final food source represents the subset of selected features.

The OA in a conventional wrapper feature selection method aims to minimise the error between the predicted output and the actual output by choosing the most effective individual features. This minimising problem is formulated as follows:

Minimise

$$Error = \sqrt{\frac{1}{T}\sum_{i}^{T}(A_i - E_i))^2} \quad (4)$$

Subject to:

$$\mathrm{E(X)} = \mathrm{F(LW \times (F(IW \times X + b_1)) + b_2)} \quad (5)$$

$$X(W) = [w_1x_1, w_2x_2, \ldots, w_mx_m] \quad (6)$$

$$0 \leq w_k \leq 1 \quad (7)$$

$$w_k \in N \quad (8)$$

where E_j, A_j, and T are the ANN's predicted output, actual values of the output, and the number of data records in each dataset, respectively. Moreover, F, LW, IW, b_1, b_2, and m are the ANN's activation function, hidden layer's weights, input layer's weights, input layer's bias, hidden layer's bias, and the number of potential features in the candidate features' pool, respectively. X is the candidate features' pool vector, w_k indicates whether x_k (*k-th* feature) is selected or not. The first OA in OA2DD, on the other hand, divides the set candidate features into different subsets. The elements of each subset have a temporal

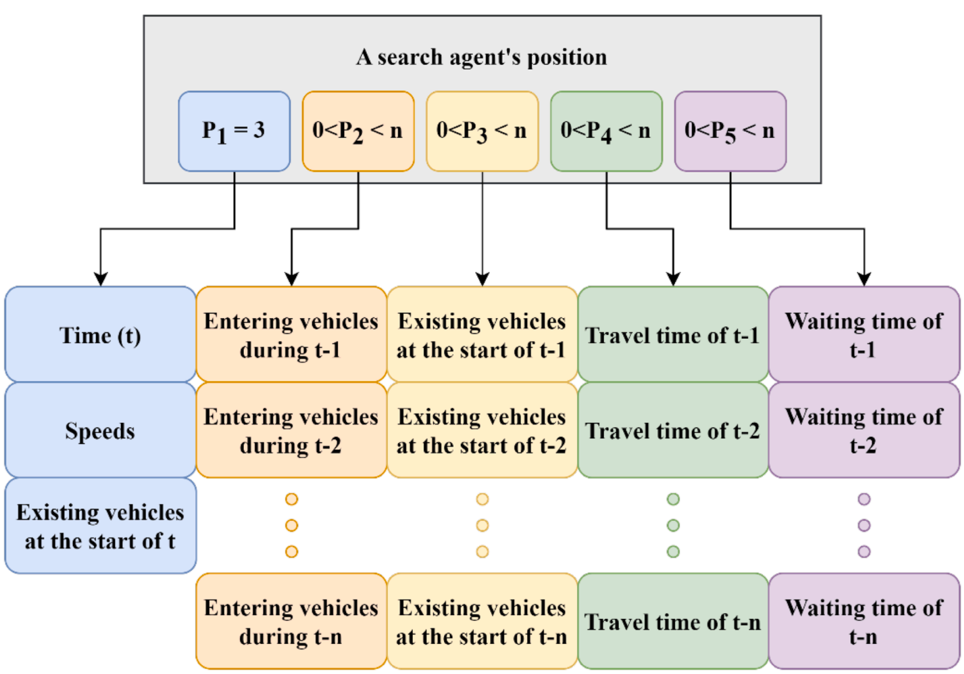

Fig. 6. The definition of a search agent's position in the first OA.

correlation with each other and are sorted from the most recent value to the oldest. The first OA's next step is minimising the prediction error by selecting several sequential elements from each subset. This minimising problem is the same as Eq (4). However, it is subjected to different conditions. The subjects are formulated as follows:

Subject to:

$$E(X) = F(LW \times (F(IW \times X + b_1)) + b_2) \tag{9}$$

$$x_1, x_2, \ldots, x_n \subset X \tag{10}$$

$$x_j(z) = \{z_j x_{j,1}, (z_j - 1)x_{j,2}, \ldots, (z_j - m + 1)x_{j,m}\} \tag{11}$$

$$0 \le z_j \le m \tag{12}$$

$$z_j \in N \tag{13}$$

where x_j is the *j-th* subset of candidate features and z_j indicates the number of selected features from the *j-th* subset of the set of candidate features.

The main difference between the first OA and a conventional wrapper feature selection method is in their minimising functions. The first OA solves Eq. (4), while a conventional wrapper feature selection solves Eq. (5). These equations are written assuming that an ANN with only one hidden layer is used as a DDPM and the error is calculated by root mean square error in both methods.

A conventional wrapper feature selection method treats all features equally without considering any relation among them. This means that it can select, for example, the number of entering vehicles during the *t*-9*th* time interval as a selected feature without selecting the same value from any other time interval. On the contrary, the first OA in OA2DD selects that value if and only if the same value from all the time intervals between *t*-9 and *t* is selected. b) Hybridising the second OA and the DDPM

As it was mentioned before, in the first OA, to calculate each agent's fitness, the second OA is called. The second OA's goal is to find the optimum adjustment parameters for the DDPM for the current subset of features. Its general process is similar to the first OA except for the definition of agents' position. After determining the initial parameters, it generates the agents randomly. Each agent's position represents a different set of adjustment parameters for the DDPM. In this study ANN is used as the DDPM, however, this logic can be applied to any other DDPM. As shown in Fig. 7, in the second OA, a search agent's position consists of two parts, i.e., *L* and *N*. Part *L* is binary and indicates whether a layer is active or not while Part *N* determines the number of neurons in each of the hidden layers. The maximum number of hidden layers, i.e., *m*, and the maximum number of neurons in each layer, i.e., *k*, should be preset by the user based on the complexity of the problem. For instance, when *m* and *k* are 3 and 20 respectively, an agent with the position of [1,0,1,4,7,2] represents an ANN that has two hidden layers (the second hidden layer is deactivated) with 4 and 2 neurons in them respectively. In the next step, the second OA calls the DDPM to calculate each agent's fitness. Then, in each iteration, the best agent is compared to the food source and replaces it if has a lower error. Subsequently, the position of each agent is updated, and the next iteration starts and goes through the same process until the last iteration. The final food source represents the optimum structure of the DDPM, i.e., ANN, with the lowest prediction error for the current subset of features, i.e., the search agent from the first OA.

For every agent in the second OA, the DDPM is called once. Every time that the DDPM is called it gets the input dataset consisting of the selected features by the first OA's agent, then adapts to the adjustment parameters determined by the second OA's agent and calculates the prediction error. In the case of ANN, it needs to be trained every time that is called according to the selected features and determined structure.

3.4. Data-driven model and prediction

After selecting the features, the data-driven model is trained using the selected features to predict the mean travel time in real time. This part operates in real-time, meaning that when the model is trained, the necessary data can be collected from the road segment under study and the model predicts the mean travel time for the current time interval in almost a second. To make the feature selection and predictive model more compatible with each other, it is advised to use the hybridization of the second OA and DD as the predictive model. In this study, an ANN is used for that purpose. During the training, validation, and testing process, root mean square error (RMSE) is used as the statistical error indicator. Furthermore, to present the error of the prediction other indicators including Pearson correlation coefficient (R), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), scatter index (SI), Mean Bias Error (MBE), and RMSE are used (a test dataset is used for presenting and comparing). To do so, the results of the testing phase

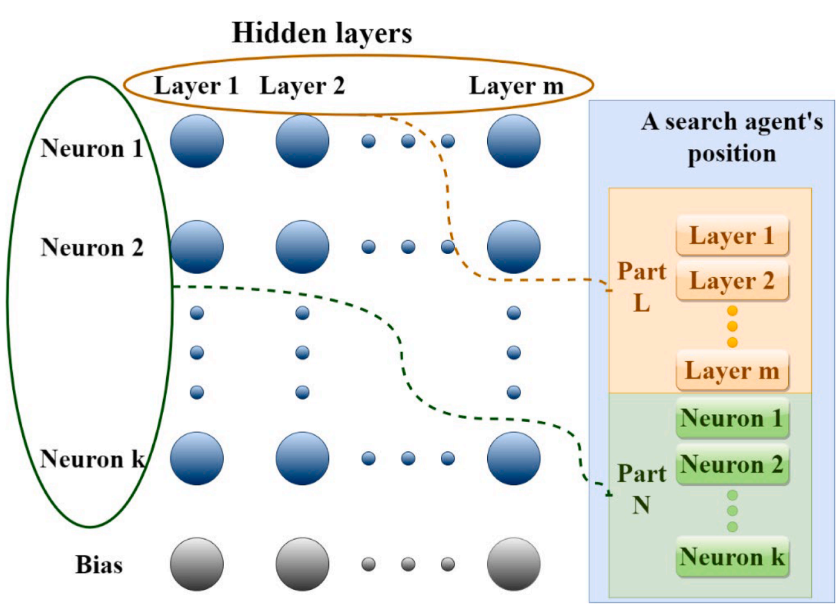

Fig. 7. The definition of a search agent's position in the second OA.

are compared.

$$R = \frac{\sum_{j=1}^{T} \left(E_j - \overline{E}\right)\left(A_j - \overline{A}\right)}{\sum_{j=1}^{T} \left(E_j - \overline{E}\right)^2 \sum_{j=1}^{T} \left(A_j - \overline{A}\right)^2} \tag{14}$$

$$RMSE = \sqrt{\frac{1}{T}\sum_{j=1}^{T} \left(A_j - E_j\right)^2} \tag{15}$$

$$MAE = \frac{1}{T}\sum_{j=1}^{T} \left|A_j - E_j\right| \tag{16}$$

$$MAPE = \frac{100}{T}\sum_{j=1}^{T} \left|\frac{A_j - E_j}{A_j}\right| \tag{17}$$

$$SI = \frac{RMSE}{\overline{A}} \tag{18}$$

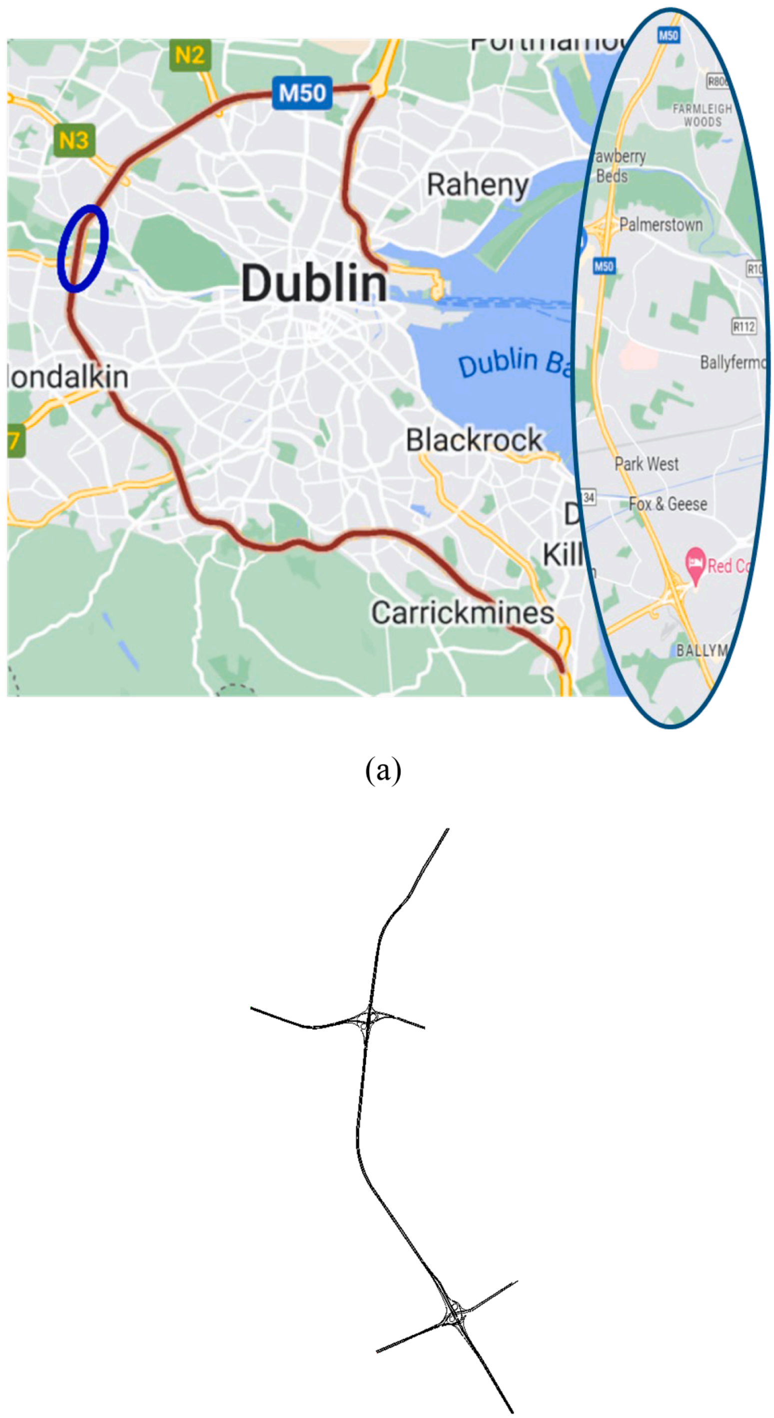

Fig. 8. M50 motorway. Source: Google Maps. (b) under examination part of M50 in the simulator.

$$\mathrm{MBE} = \frac{1}{\mathrm{T}}\sum_{j=1}^{\mathrm{T}}\left(\mathrm{E_j} - \mathrm{A_j}\right) \tag{19}$$

where E_j $\overline{E}$, A_j, $\overline{A}$, T, are the ANN's estimated output, average of estimated outputs, actual values of the output, average of actual output, the number of data records in each dataset, and the mean value of the actual travel time, respectively. SI indicates the performance of a prediction. If $SI \leq 0.1$ the performance is "excellent", if $0.1 < SI \leq 0.2$ the performance is "good", when $0.2 < SI \leq 0.3$ the performance is "fair", and if $0.3 < SI$, the performance is "poor" [73]. MBE indicates if a prediction is overestimated ($MBE > 0$) or underestimated ($MBE < 0$) and quantifies it. Finally, after training the data-driven model using the collected dataset and its selected features using OA2DD, this data-driven model can be used in the real-time process.

4. Case study: M50 motorway in Dublin

The OA2DD is demonstrated in a simulation study conducted on the M50 motorway in Dublin. The M50, which takes on a C-shape as depicted in Fig. 8(a), serves as a vital link connecting the southern and northern regions of Dublin. As Ireland's busiest motorway, congestion on the M50 has become a notable issue [74]. Existing data indicates that in 2021, there were 45.1 million passages recorded on the M50 motorway [75]. In this research, to evaluate the traffic dynamics, a detailed simulation of the motorway is carried out using the Simulation of Urban MObility (SUMO) tool [76]. Fig. 8(a) also depicts the particular segment being analysed on the M50 map, while Fig. 8(b) displays the same segment within SUMO showcasing five lanes and maintaining the standard speed limit of 100 km/h. It is essential to highlight that this study exclusively examines the southbound direction of the M50.

The M50 model utilized in SUMO for this paper was developed by Gueriau and Dusparic [77]. This model integrates both Human Driving Vehicles (HDV) and Connected and Autonomous Vehicles (CAV) operating at automation levels 2 and 4, denoting automation with human supervision and fully automated driving, respectively [77]. The researchers designed six distinct scenarios encompassing various ratios of HDVs and CAVs. For this study, Scenario A is selected, which exclusively involves 100 % HDVs with no CAVs. Demand patterns and volumes were determined using observed data, averaging daily observations from 2012 to 2019. However, since this dataset only includes the same speed limit for every lane, i.e., 100 km/h, in this study, the dataset from the mentioned study is imported to SUMO and by changing each lane's speed limits to different values between 36 km/h and 144 km/h a bigger dataset is produced to study the impact of per-lane speed limits on travel time as input features. Then, the dataset is randomised, the first thousand records are used in feature selection (Section 5.3), and the second thousand records are used to train the DDPM (Section 5.4). In both cases, the dataset is divided into a training set (80 %), a validating set (10 %), and a testing set (10 %). Moreover, the collected observed dataset [77] is used for real-time prediction (Section 5.5). Table 2 presents the statistical values of travel time in each dataset.

Table 2
Statistical values of travel time (s) in different datasets.

Statistical values	Feature selection dataset	Data-driven dataset	Real-time dataset
Minimum	229.69	231.2	244.33
Maximum	1067.3	1137.7	969.6
Mean	512.4	505.2	456.5
Standard deviation	234.62	230.52	237.92
Skewness	0.48	0.51	0.82
Kurtosis	1.80	1.81	2.00

5. Results and discussion

5.1. Feature extraction

As mentioned before, the first step for extracting features is to divide the day into time intervals. For this case study, 10 min time intervals are chosen. Then, to collect the volume data, detectors are placed (Fig. 9). The number of vehicles entering the under-study area during each time interval (EC) and the number of vehicles existing in the area at the start of each time interval (IC) can be collected using these detectors. Moreover, since, the dataset used for the simulation contained the average values of seven years of data collection, the only time value that can be used is the TOD. The time of the week, the month, and the year are not available in this dataset. Regarding the speed, per-lane mean speeds are not available and cannot be applied in this case study. However, the simulation was run multiple times with different speed limits for each individual lane, so the effects of speed limits can be studied. Hence, per-lane speed limits (S) are extracted as well. Finally, the mean values of travel time and waiting time of each time interval are extracted.

5.2. Candidate features pool

Once the main features are extracted for each time interval, it is needed to determine the maximum number of previous time intervals that need to be considered in the pool. In this case study 10 previous time intervals are considered in the pool. As presented in Table 3, the candidate features pool consists of different columns (each column is allocated to a time interval). The current time interval's column includes TOD at the start of the time interval, the per-lane speed limits, the number of vehicles existing in the under-study area for the previous time intervals, the number of vehicles entering the area during the time interval, the number of vehicles already existing in the area at the start of the time interval, and the mean value of travel time and waiting time of the vehicles that exit the area during the time interval are considered as the candidates.

5.3. Feature selection

The next step is to apply the OA2DD to the candidate features pool. To compare the results of OA2DD, a normal wrapper feature selection using binary SSA with a predefined ANN (SSAANN), a Particle Swarm Optimisation (PSO) wrapper feature selection with a predefined ANN (PSOANN), an SSA wrapper feature selection with SVR (SSASVR), a PSO wrapper feature selection with SVR (PSOSVR), and a wrapper feature selecting method using binary SSA with an ANN that optimised its structure using another SSA (SANSA) are implemented on the candidate features pool. The only difference between OA2DD and SANSA is that the first OA in SANSA is a binary OA and does not follow any instruction, such as the one that OA2DD follows to select features. Moreover, because of the random start points of OAs and ANN, all the models are run 10 times, and the best results are presented.

Before implementing the OA2DD, its adjustment parameters are defined. These parameters are chosen with respect to the computational budget (see Table 4). The other methods utilized for comparison use the same values as Table 4 when applicable.

During the feature selection in the first OA, 2000 (number of iterations $\times$ number of search agents) individual search agents go through a search area of $11^4 = 14641$ different combinations to achieve the 12 selected features. This number of combinations is calculated based on the fact that there are five columns in the candidate features' pool, the features in the first columns are always chosen, and the potential number of selected features from any of the other four columns is between 0 and 10.

For each one of those search agents from the first OA, another set of 2000 individual search agents go through a search area of $20^3 = 8000$ (three layers with 20 potential neurons in each) different structures to

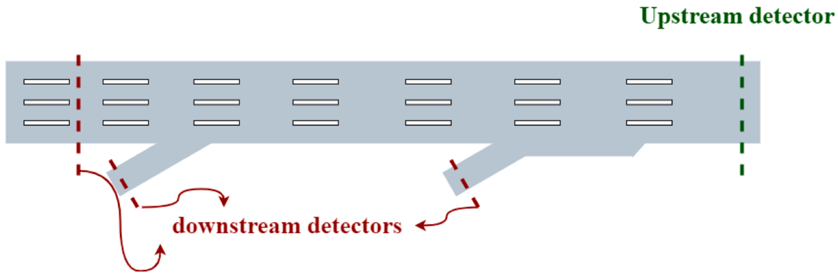

Fig. 9. Schematic locations of the detectors.

Table 3
Case study's candidate features.

t (current time interval)	t-1	t-2	…	t-10
Time of the day	Entering vehicles	Entering vehicles		Entering vehicles
Lane1 speeds limit	Existing vehicles	Existing vehicles		Existing vehicles
Lane2 speeds limit	Mean travel time	Mean travel time		Mean travel time
Lane3 speeds limit	Mean waiting time	Mean waiting time		Mean waiting time
Lane4 speeds limit				
Lane5 speeds limit				
Existing vehicles				

Table 4
OA2DD adjustment parameters.

Parameter	Value
Maximum number of iterations	100
The number of search agents	20
Maximum number of hidden layers	3
Maximum number of neurons in each hidden layer	20
Activation function in hidden layers	Hyperbolic tangent sigmoid
ANN training algorithm	Levenberg-Marquardt
Cost function	RMSE
The number of total runs	10

find the optimum structure for ANN with respect to the first OA's agent position. The comparison feature selection methods are also applied to the case study and Fig. 10 compares their convergence curves to OA2DD's. As can be seen, the proposed OA2DD has the best performance among the others followed by SANSA which is the binary version of OA2DD. Furthermore, SSAANN, SANSA, and SSASVR achieve their minimum fitness value (RMSE) quicker than the other methods with a few noticeable drops while OA2DD, PSOANN, and PSOSVR's fitness value gradually decrease. Therefore, it can be concluded that OA2DD do not fall into local optimum, because of its thorough search.

Moreover, Fig. 11 presents the selected features by each feature selection method. As can be seen, one of the advantages of OA2DD compared to conventional wrapper feature selection methods is that the features are selected following a structure from the closest time intervals to the current one. The furthest selected features by OA2DD are from the fourth time interval before the current one. The other methods, on the other hand, selected at least one feature from the 10th time interval, while not choosing features from more recent time intervals. Hence OA2DD makes the data collection easier and cheaper. Furthermore, OA2DD selected 19 features which is less than the selected features by the other methods (22-25). This advantage of OA2DD reduces the cost of data collection as well as computational cost. The complexity of an ANN which is defined as the sum of the number of links in the ANN, i.e. the number of weights and biases, is used as the computational cost (CC) indicator in this study [78,79] and is calculated as follows:

$$CC = \sum_{i=1}^{I} (N_i(N_{i-1} + 1)) \tag{20}$$

where CC is the complexity, N_i is the number of neurons in the ith layer, N_1 and N_I are the number of neurons in the input and output layer, respectively, and I is the number of layers in the ANN. For example, for an ANN with 5 inputs, one hidden layer with 7 neurons in it, and one output CC calculates as follows:

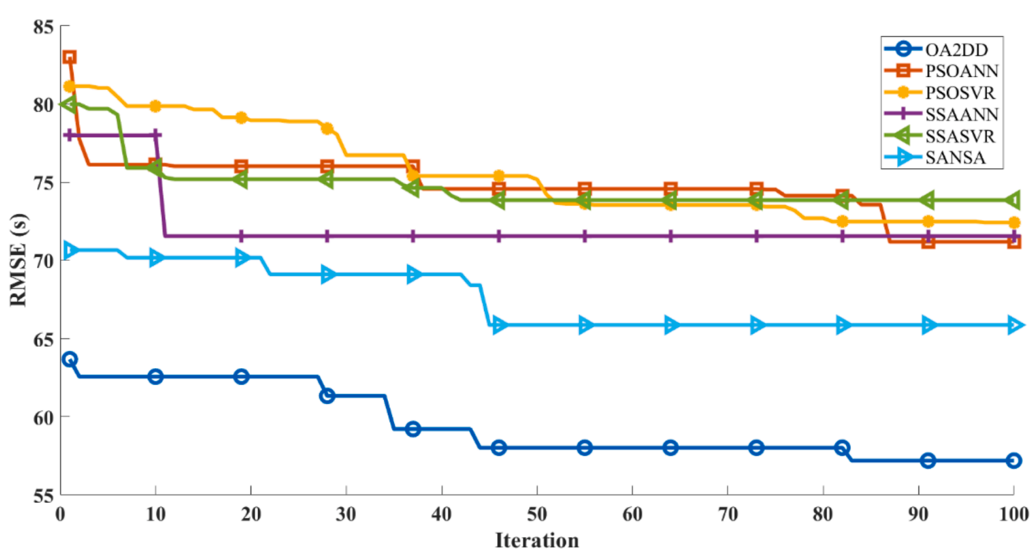

Fig. 10. Convergence curves of six different feature selection algorithms.

	OA2DD	SANSA	PSOANN	PSOSVR	SSAANN	SSASVR
TOD	X	X	X			
S_1	X		X	X	X	
S_2	X		X	X		
S3	X	X				
S4	X	X	X		X	X
S5	X	X	X	X		
IC	X	X	X	X	X	X
EC_{t-1}	X	X	X	X	X	X
ECt-2	X	X	X		X	
EC_{t-3}	X	X				
ECt-4	X			X		X
EC_{t-5}			X			X
EC_{t-6}		X		X		
ECt-7			X			
EC_{t-8}			X			X
ECt-9			X	X	X	
ECt-10		X		X		
IC_{t-1}	X			X		X
ICt-2	X				X	
IC_{t-3}	X	X		X		X
ICt-4			X			X
IC_{t-5}		X				X
IC_{t-6}		X	X		X	
ICt-7					X	X
IC_{t-8}		X	X		X	X
ICt-9					X	
ICt-10				X		X

	OA2DD	SANSA	PSOANN	PSOSVR	SSAANN	SSASVR
MT_{t-1}	X				X	
MTt-2	X		X		X	X
MT_{t-3}		X			X	
MTt-4		X	X			
MT_{t-5}				X	X	X
MT_{t-6}			X	X	X	X
MTt-7		X			X	X
MT_{t-8}					X	
MTt-9		X		X	X	X
MTt-10				X	X	X
MW_{t-1}	X					
MWt-2	X	X	X	X		X
MW_{t-3}	X		X	X		
MWt-4		X	X	X		X
MW_{t-5}		X		X	X	X
MW_{t-6}			X		X	
MWt-7		X				
MW_{t-8}				X	X	X
MWt-9			X			
MWt-10			X			
Number of selected features	19	24	25	22	24	24

X: shows the features selected by each method.

Fig. 11. Selected features by six different feature selection methods.

$$CC = (7 \times (5+1)) + (1 \times (7+1)) = 38 \tag{21}$$

Moreover, if an ANN with 10 neurons in its hidden layer is trained with the selected features by all the mentioned methods and used to predict the travel time in real-time, the architecture of the ANNs that are trained by selected features by any of the mentioned methods and their computational costs are presented in Table 5.

5.4. Data-driven model

The second OA's job in OA2DD is to optimise the structure of the ANN according to the selected features by the first OA. The optimised ANN according to the final selected features has a hidden layer with 10 neurons in it. The same structure is used for the data-driven model in the rest of this study. As mentioned in Section 3.4, the DDPM is trained using a data set with the selected features. In this section, an ANN with a hidden layer with 10 neurons in it is chosen as the DDPM, and it is trained on a new dataset (see Section 4) to predict the travel time given each subset of features and the full set of features. Table 6 presents different testing error indicators to compare the performances of ANNs trained by different subsets of features.

OA2DD's RMSE by the value of 49.13 seconds is lower than SANSA's by 11 %, the full set's by 29 %, PSOANN by 12 %, PSOSVR by 29 %, SSAANN by 12 %, and SSASVR by 17 %. Pearson correlation coefficient (R), between 0 and 1, indicates how the predicted mean travel times match their actual values, and as it can be seen all the feature subsets provide an R equal or higher than 0.95 where OA2DD has the highest

Table 5
Reduction in computational cost resulted from feature selection.

Method	Selected Features	ANN's architecture	CC	Reduction in CC
Full set	47	47-10-1	481	0
OA2DD	**19**	**19-10-1**	**211**	**56 %**
SANSA	24	24-10-1	261	46 %
PSOANN	25	25-10-1	271	44 %
PSOSVR	22	22-10-1	241	50 %
SSAANN	24	24-10-1	261	46 %
SSASVR	24	24-10-1	261	46 %

Table 6
Errors of an ANN predicting mean travel time using features subsets selected by different methods. The best results are highlighted in boldface.

Feature subset	RMSE (s)	R	MAE (s)	MAPE (%)	SI	MBE (s)
OA2DD	**49.13**	**0.98**	**34.18**	**6.64**	**0.10**	1.20
SANSA	55.48	0.97	38.78	7.58	0.11	2.5
PSOANN	56.16	0.97	41.49	8.54	0.11	3.58
PSOSVR	69.43	0.95	48.88	9.9	0.14	2.45
SSAANN	56.00	0.97	41.83	8.61	0.11	**0.54**
SSASVR	59.31	0.97	45.16	9.63	0.12	3.15
Full set	69.33	0.95	50.09	9.87	0.14	2.17

value with 0.98, which shows a high correlation between the predicted travel time and the calculated travel time by SUMO. Moreover, using the feature subset selected by OA2DD has the least MAE and MAPE values, i.e., 34.18 seconds and 6.64 %, respectively, and according to its MAPE value, it has the accuracy of 93.46 % which is the highest amongst the others. On the other hand, the ANN trained by the full set of features has the least accuracy by having the same values of 50.09 seconds and 9.87 %. According to SI, using selected features by OA2DD results in an "excellent" performance while the other methods provide a "good" performance, and according to MBE, all the methods result in an overestimation of mean travel time, OA2DD providing the second best score. Overall, using OA2DD results in the most accurate prediction among all the methods, while using the full set of features results in the least accurate prediction.

The better performance of SSAANN compared to SSASVR, and PSOANN compared to PSOSVR shows the importance of compatibility between the DDPM used in the wrapper feature selection algorithm and the one used as the predictor. Moreover, the superiority of OA2DD to SANSA proves that the rationale of the first OA in OA2DD that considers the temporal correlation amongst input features, results in not only easier and cheaper data-collection with lower computational costs but also in more accurate prediction. Real-time prediction

After developing the DDPM, it can be implemented and used. In this section, the ANNs developed, trained, and tested in the previous section using different feature subsets are implemented on the original observed dataset (see Section 4) consisting of 133 records. Table 7 presents and compares the performances of ANNs trained by each feature subset in real-time prediction.

Results show that PSOANN provides the minimum RMSE with a value of 57.11 followed by OA2DD by only a 2 % difference, and SANSA is the third one by 3 %. Moreover, PSOANN has slightly better R and SI compared to OA2DD. On the other hand, the MAE and MAPE of OA2DD are the overall lowest and lower than the same values of PSOANN by 17 % and 25 % respectively. Comparing OA2DD with other modes indicates that it has a lower RMSE, lower or equal R, lower MAE, and MAPE. However, in the case of SI, SANSA and PSOANN perform slightly better than OA2DD, and OA2DD comes in fifth place in the case of MBE.

5.5. The impact of using different optimisation algorithms

The proposed methodology in this study consists of two OAs and SSA is used to showcase the methodology performance. However, any other OAs can also be used instead of SSA. Therefore, three other commonly used OAs in feature selection are also used in the same methodology. These algorithms include PSO, Grey Wolf Optimisation (GWO) [80], and Whale Optimisation Algorithm (WOA) [81] The same dataset that was used for the feature selection phase (see Section 5.3) is given to the other models and their convergence curves are presented in Fig. 12. All the methods were run 10 times, and their best performances are presented. It should be noted that the model named OA2DD here is the same one in previous sections and it uses SSA as the OA. However, its name is not changed in this section to avoid confusion. As can be seen in Fig. 12, all models perform similarly. They all have a big reduction in error around

Table 7
Errors of an ANN predicting mean travel time using feature subsets selected by different methods for real-time prediction. The best results are highlighted in boldface.

Feature subset	RMSE (s)	R	MAE (s)	MAPE (%)	SI	MBE (s)
OA2DD	58.38	0.97	**37.64**	**8.14**	0.13	5.00
SANSA	58.72	0.97	40.98	9.00	**0.12**	5.2
PSOANN	**57.11**	**0.98**	45.43	10.80	**0.12**	6.67
PSOSVR	61.99	0.97	42.34	8.54	0.14	**2.81**
SSAANN	61.17	0.97	43.90	9.28	0.13	**2.81**
SSASVR	65.61	0.96	45.48	9.82	0.14	4.56
Full set	66.92	0.96	43.65	9.19	0.14	3.67

the same area, i.e., between the 20th and 40th iterations and another decrease between the 80th to 100th iterations. Moreover, the difference between the most accurate method, i.e., WOA-OA2DD and the least accurate method, i.e., GWO-OA2DD is less than two seconds. Overall, OA2DD with any metaheuristic OA performs better than the conventional wrapper feature selection shown previously.

5.6. Explanatory analysis

One of the main reasons for predicting traffic is to help decision-makers to be able to adapt the traffic conditions to the current situation and reduce congestion with methods such as variable speed limits and ramp metering. It can also be used to inform the road users, so they can avoid congestion by changing their route or departure time. Therefore, the accuracy of travel time prediction in congested traffic can be argued as more important than that of free-flow conditions. A comparison of actual mean travel time and predicted mean travel time using the subset of features selected by OA2DD is illustrated in Fig. 13(a). As can be seen, the predictor performs well throughout the day even in congested traffic. The longest period of time that the predictor is not able to match the real value consistently is between 11:00 and 15:00. This phenomenon is due to the sensitivity of the predictor to EC. As shown in Fig.12(b), before 6:00 and after 21:00 less than 2000 cars per hour enter the road, and consequently the road is under free-flowing conditions. Then EC and IC start to rise up to the period from 11:00 to 15:00 where around 5000 cars per hour still enter the road, but the road goes back to free-flow condition because of lower IC. Due to the rise in the EC, the predictor expects the mean travel time to be higher than what it actually is. The trends in other input parameters such as the IC, that is more compatible with mean travel time, as shown in Fig. 13(b), help to increase the accuracy of those parts.

As observed, an increase in both IC and EC starting around 5:00 leads to a rise in mean travel time. Both IC and travel time reach their peak around 9:00. After 9:00, due to the earlier decrease in EC and the current decline in IC, mean travel time begins to decrease as well. Between 11:00 and 15:00, despite relatively high EC, the sharp drop in IC results in a continued reduction in mean travel time. The pattern during the second peak mirrors that of the first. Additionally, Fig. 13 shows that when IC and EC are either both high or low, the prediction accuracy improves. Conversely, accuracy decreases when IC and EC have opposing values, i.e., when one is high and the other is low.

Moreover, in predicting traffic, if there is a prediction error, operationally it is better to see a false congestion rather than a false free flow. In the case of false free-flow decision-makers lose the chance to react to the situation and cannot adapt the road to the traffic. As can be seen in Fig. 13, when there is a relevant error in the predicted travel time, the predictor overestimates the travel time except for 8:00 and 19:00. Thus, that error does not keep the decision makers from reacting. This overall overestimating can also be seen in MBE. Furthermore, as can be seen, the predictor is able to catch the trend and follow the changes in mean travel time throughout the day. This ability of the predictor, helps the decision-makers to be informed about the upcoming congestion and to have the chance to adapt the road to it before it happens. Nonetheless, this specific sensitivity of the prediction with the decrease of the IC can be the aim of future research as emphasises an important trend in the dynamics of traffic for effects of prediction.

5.7. Model interpretation

The predicted travel times derived from traffic data can be utilized to assess traffic conditions and improve route planning. For these applications, ensuring that the model is interpretable is essential, so traffic engineers and planners can rely on the insights generated by the ML model in their routine tasks. In particular, this approach could provide in-depth information about which features are most influential when predicted travel times differ significantly from usual patterns. To do so,

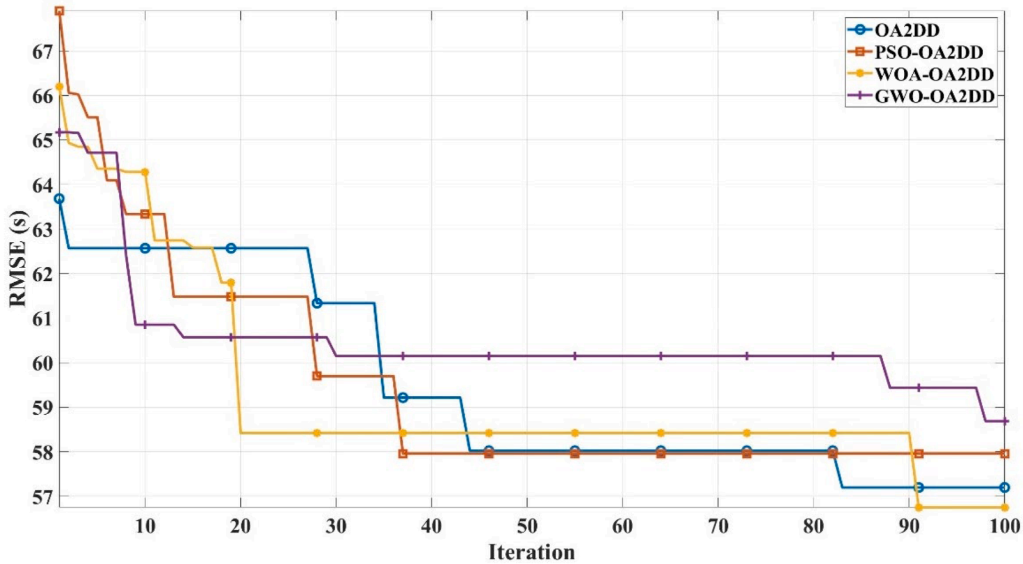

Fig. 12. OA2DD's convergence curves using different OAs.

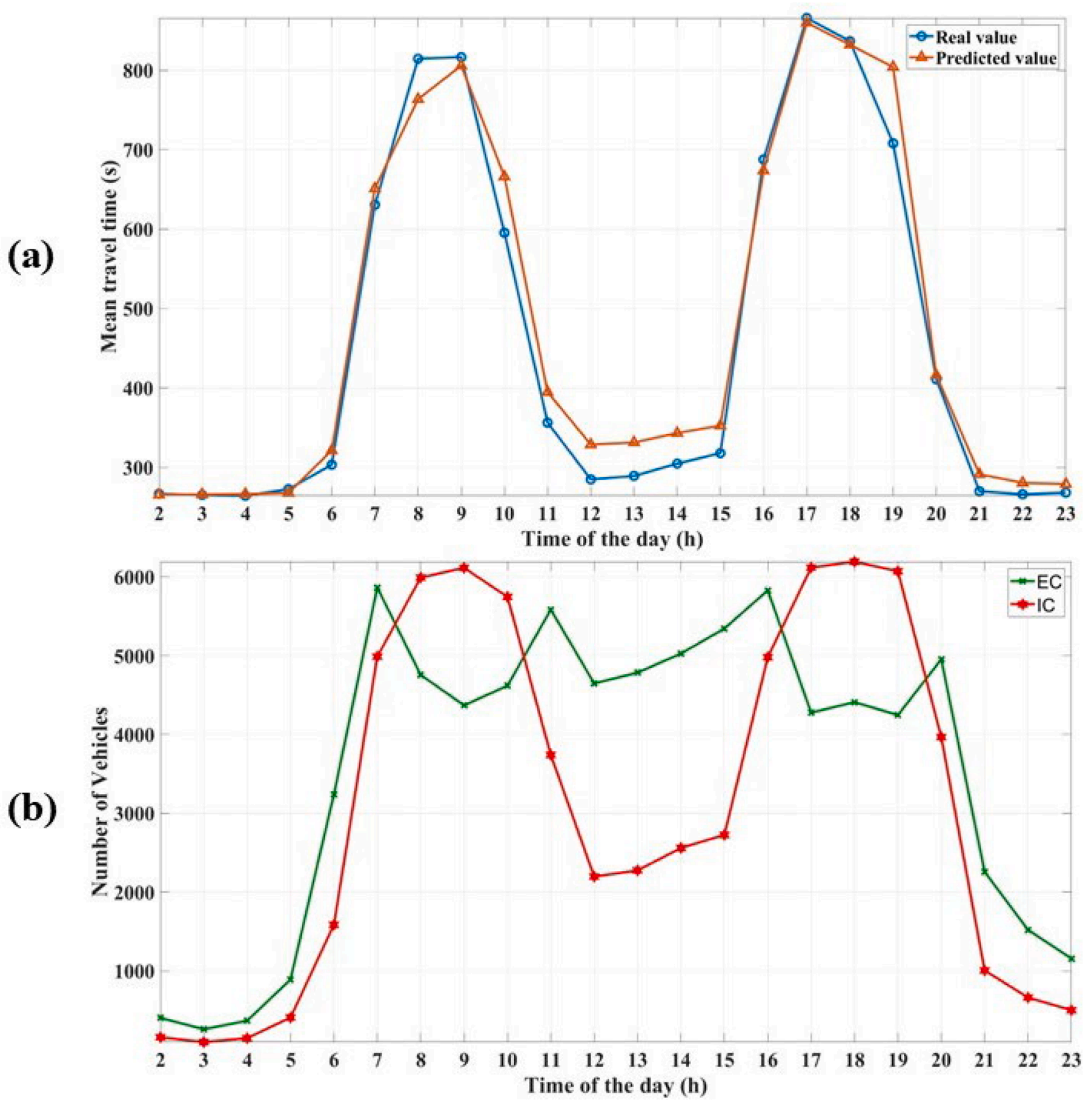

Fig. 13. (a) Real mean travel time vs predicted mean travel time, and (b) hourly EC and IC.

Shapley Additive Explanation (SHAP) [82] is used in this paper.

SHAP leverages game theory principles to interpret the results of any ML model by determining Shapley values. These values provide insight into why a particular prediction deviates from the model's average output and quantify the contribution of each variable to the final result. The Shapley value represents the average impact of a feature across all potential combinations of features [83]. For each combination, the contribution is assessed by comparing the predicted outcome with and without the inclusion of the specific feature. In other words, The SHAP value represents the impact of a feature on the difference between the predicted value for a specific query point and the average prediction. For each query point, the total deviation from the average prediction is equal to the sum of the SHAP values across all features. Fig. 14 shows the SHAP figure. The SHAP value for the *a-th* feature and the query point *q* is defined by the value of *v*:

$$\varphi_a(v_q) = \frac{1}{B}\sum_{S\subseteq\beta\{a\}} \frac{v_q(S\cup\{a\}) - v_q(S)}{\frac{(B-1)!}{|S|!(B-|S|-1)!}} \tag{22}$$

where B is the number of all features, β is the set of all features, $|S|$ is the number of elements in set S,

As shown in Fig. 14(a), IC has the most significant impact on the predicted mean travel time, with a Shapley value exceeding 180, while the second most influential feature is TOD, with a Shapley value over 20. The high Shapley value of IC accounts for the similarity between its daily trend and that of the mean travel time. Additionally, as seen in Fig. 14 (b), except for IC, which spans a wide range on both the positive and negative sides of the graph, the other features are more concentrated around 0. Furthermore, it is evident that some features, such as IC_{t-2} and EC_{t-4}, have a reverse impact on the predictor. This means that as their values increase, the predictor value decreases.

6. Limitations and future opportunities

There are some limitations in this study and some opportunities that need to be explored in the future. First, the same rationale can be applied to feature selection and prediction for different traffic parameters, such as traffic flow and mean speed. Then, the effect of the size of the road segment and time intervals as well as mixing HDVs with CAVs on the selected features needs to be studied. Moreover, the transferability of the method, specifically, whether the features selected for the M50 are also the best choices for another highway, needs to be investigated. As mentioned in the paper, the day of the week, the week of the month, and the month of the year have important impacts on traffic and travel time. However, in this paper, the case study focused on 24 h traffic. Larger periods can also be explored in further studies. A more comprehensive dataset that includes more information such as weather and road surface conditions can also increase the accuracy of the prediction. This

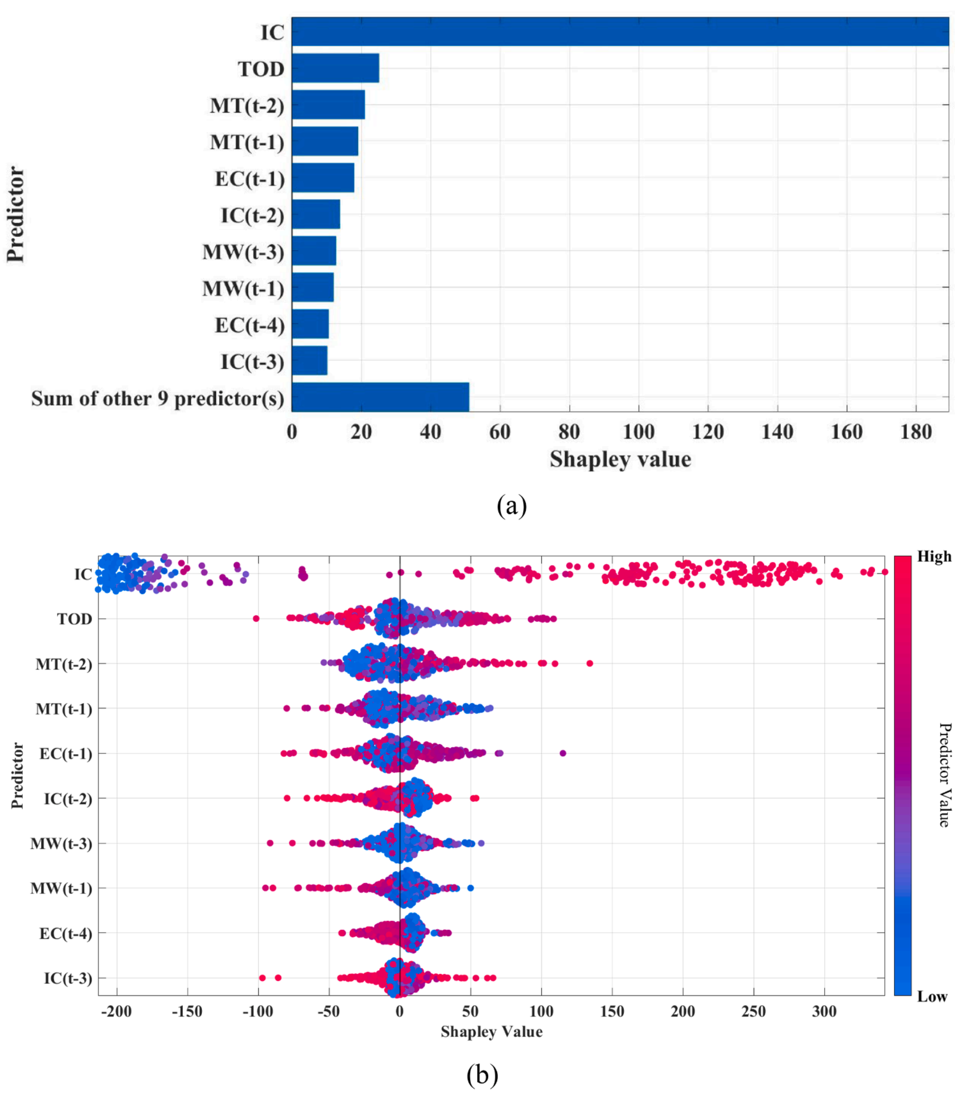

Fig. 14. SHAP values. (a) Global feature importance, (b) Local explanation summary.

information was not available for this research. Moreover, the simulation used in this study was built based on a dataset from 2012 to 2019. More recent datasets can enhance the reliability of this method. Nonetheless, it is highlighted that within this context the OA2DD performed consistently with a prediction accuracy of less than 8.14 % in relative error.

It is worth noting that the current methodology relies on data during the training phase, prompting discussion on where such data can be sourced within the traffic system. The assumption made in this study is that much of this data either already exists or will become readily available in the near future, given the significant data collection efforts in traffic systems (e.g., tolls, vehicle counters). Additionally, data from vehicle users, such as cell data or connected vehicles, could prove valuable for further investigations. The availability of such datasets would provide all necessary data for this implementation. However, the presented methodology and rationale can be implemented on any kind of existing dataset. In conclusion, it is crucial to emphasize that this study aims to highlight how OA2DD concepts can significantly impact travel time prediction, particularly through the integration of data and both offline and online training for real-time operation.

Moreover, in this study, because of the nature of the case study and data accessibility external data such as the weather and road surface conditions were not considered. The case study was a 24 h simulation that was used as a proof of concept. However, considering the external data and how they should be imported into OA2DD is an opportunity for further exploration. In this regard, questions such as, would a new feature that expands the space be required, or will the same set of features in the ANN suffice or is there a need to define scenario-dependent ANN that are concurrent, need to be answered in future research.

7. Conclusion

This study introduces a comprehensive methodology, which encompasses the entire process from data collection to travel time prediction. Notably, the introduced methodology incorporates a novel feature selection method that considers temporal correlations between input features and the output variable, aiming to enhance the accuracy of travel time prediction. The implementation of the OA2DD is examined through a case study utilizing the M50 motorway in Dublin, with SUMO employed for training, validating, and testing its outputs. The scenarios used to train the OA2DD were constructed in SUMO, drawing from an observed dataset. The following points are concluded from the study.

- The hybridization of two OAs and a DDPM (OA2DD) can reduce the computational cost in travel time prediction by reducing the number of selected features compared to standard wrapper feature selection methods.
- An ANN trained with selected features by OA2DD outperforms ANNs trained with selected features by other wrapper methods across all statistical indicators.
- Using the full set of features to train an ANN, results in the least accuracy and highest computational cost in the prediction which means feature selection is a necessary step before the prediction of travel time.
- The fact that the ANN trained by the features selected by PSOSVR and SSASVR are the second and third least accurate ones, respectively, shows that the compatibility of the DDPM of the feature selection method and prediction method is important.
- The difference between OA2DD's performance and SANSA's performance shows the introduced rationale of selecting features with respect to their temporal correlation provides more accuracy with less computational cost.
- According to the results, for predicting mean travel time for 10 min time intervals, the past four time intervals are the ones that have a relevant impact on the prediction.

Moreover, the findings of this study have several policy implications including the following:

- The prediction model can enhance real-time traffic management by allowing authorities to adjust signals and lanes proactively, reducing congestion during peak hours.
- Accurate travel time predictions help policymakers target infrastructure improvements, ensuring efficient allocation of resources to reduce congestion.
- The model can inform policies like congestion pricing, encouraging alternative transport modes and reducing peak-time congestion.
- While the benefits of this approach are significant, it is essential that its implementation runs in parallel with policies ensuring the safe usage of AI, where techniques like exploratory analysis play a critical role in maintaining safety and effectiveness.

CRediT authorship contribution statement

Amirreza Kandiri: Writing – original draft, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Ramin Ghiasi:** Writing – original draft, Methodology. **Maria Nogal:** Writing – review & editing, Supervision. **Rui Teixeira:** Writing – review & editing, Supervision, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] M. Shaygan, et al., Traffic prediction using artificial intelligence: review of recent advances and emerging opportunities, Transport. Res. Part C: Emerg. Technol. 145 (2022) 103921.
[2] M. Elassy, et al., Intelligent transportation systems for sustainable smart cities, Transport. Eng. (2024) 100252.
[3] X. Zhang, Review on Traffic Flow Prediction Method Based on Neural Network, in: 2023 2nd International Conference on Artificial Intelligence and Autonomous Robot Systems (AIARS), IEEE, 2023.
[4] S.R. Vankdoth, M. Arock, Deep intelligent transportation system for travel time estimation on spatio-temporal data, Neural. Comput. Appl. 35 (26) (2023) 19117–19129.
[5] H. Gao, et al., Synchronized entry-traffic flow prediction for regional expressway system based on multidimensional tensor, Transp. Res. Rec. 2675 (10) (2021) 291–302.
[6] E. Castillo, et al., A Bayesian method for estimating traffic flows based on plate scanning, Transportation 40 (1) (2013) 173–201.
[7] I.O. Olayode, L.K. Tartibu, M.O. Okwu, Prediction and modeling of traffic flow of human-driven vehicles at a signalized road intersection using artificial neural network model: A South African road transportation system scenario, Transport. Eng. 6 (2021) 100095.
[8] P. Cai, et al., A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting, Transport. Res. Part C: Emerg. Technol. 62 (2016) 21–34.
[9] K. Guo, et al., Optimized graph convolution recurrent neural network for traffic prediction, IEEE Trans. Intell. Transp. Syst. 22 (2) (2020) 1138–1149.
[10] J. Jiang, et al., Modelling traffic flows and estimating road travel times in transportation network under dynamic disturbances, Transportation 47 (2020) 2951–2980.
[11] F. Zhou, et al., Reinforced spatiotemporal attentive graph neural networks for traffic forecasting, IEEE Int. Things J. 7 (7) (2020) 6414–6428.
[12] J. James, Citywide traffic speed prediction: a geometric deep learning approach, Knowledge-Based Syst. 212 (2021) 106592.
[13] T. Ma, Z. Zhou, B. Abdulhai, Nonlinear multivariate time–space threshold vector error correction model for short term traffic state prediction, Transport. Res. Part B: Methodol. 76 (2015) 27–47.
[14] L. Moreira-Matias, F. Alesiani, Drift3flow: Freeway-incident prediction using real-time learning, in: 2015 IEEE 18th international conference on intelligent transportation systems, IEEE, 2015.

[15] H. Li, Q. Wang, W. Xiong, New model of travel-time prediction considering weather conditions: case study of urban expressway, J. Transport. Eng., Part A: Syst. 147 (3) (2021) 04020161.
[16] H. Taghipour, A.B. Parsa, A.K. Mohammadian, A dynamic approach to predict travel time in real time using data driven techniques and comprehensive data sources, Transport. Eng. 2 (2020) 100025.
[17] Yuan, H., et al. Effective travel time estimation: When historical trajectories over road networks matter. in Proceedings of the 2020 acm sigmod international conference on management of data. 2020.
[18] M. Van Den Boomen, et al., Untangling decision tree and real options analyses: a public infrastructure case study dealing with political decisions, structural integrity and price uncertainty, Construct. Manag. Econ. 37 (1) (2019) 24–43.
[19] T. Liu, et al., Predicting taxi demands via an attention-based convolutional recurrent neural network, Knowledge-Based Systems 206 (2020) 106294.
[20] J. Tang, et al., Multi-community passenger demand prediction at region level based on spatio-temporal graph convolutional network, Transport. Res. Part C: Emerg. Technol. 124 (2021) 102951.
[21] J. Guo, et al., An instance-based transfer learning model with attention mechanism for freight train travel time prediction in the China–Europe railway express, Expert Syst. Appl. 251 (2024) 123989.
[22] J. Guo, et al., A CNN-Bi_LSTM parallel network approach for train travel time prediction, Knowledge-Based Syst. 256 (2022) 109796.
[23] I. Lana, et al., Road traffic forecasting: Recent advances and new challenges, IEEE Intell. Transp. Syst. Mag. 10 (2) (2018) 93–109.
[24] J. Van Lint, S. Hoogendoorn, H.J. van Zuylen, Accurate freeway travel time prediction with state-space neural networks under missing data, Transport. Res. Part C: Emerg. Technol. 13 (5-6) (2005) 347–369.
[25] M. Bai, et al., Travel-time prediction methods: a review, in: Smart Computing and Communication: Third International Conference, SmartCom, Tokyo, Japan, Springer, 2018. December 10–12, 2018, Proceedings 3. 2018.
[26] Wan, N., et al. Prediction on travel-time distribution for freeways using online expectation maximization algorithm. in Transportation Research Board 93rd Annual Meeting. 2014.
[27] Z. Xiong, et al., A three-stage framework for motorway travel time prediction, in: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), IEEE, 2014.
[28] A. Skabardonis, N. Geroliminis, Real-time estimation of travel times on signalized arterials, Transport. Traffic Theory (2005) 387–406.
[29] S. Kolidakis, et al., Road traffic forecasting—A hybrid approach combining artificial neural network with singular spectrum analysis, Econ. Anal. Policy 64 (2019) 159–171.
[30] J. Ruiz-Aguilar, I. Turias, M. Jiménez-Come, Hybrid approaches based on SARIMA and artificial neural networks for inspection time series forecasting, Transport. Res. Part E: Logist. Transport. Rev. 67 (2014) 1–13.
[31] S. Innamaa, Short-term prediction of travel time using neural networks on an interurban highway, Transportation 32 (2005) 649–669.
[32] C.-H. Wu, J.-M. Ho, D.-T. Lee, Travel-time prediction with support vector regression, IEEE Trans. Intell. Transp. Syst. 5 (4) (2004) 276–281.
[33] M. Castro-Neto, et al., Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions, Expert Syst. Appl. 36 (3) (2009) 6164–6173.
[34] P. Gao, et al., Travel time prediction with immune genetic algorithm and support vector regression, in: 2016 12th World Congress on Intelligent Control and Automation (WCICA), IEEE, 2016.
[35] P.-Y. Ting, et al., Freeway travel time prediction using deep hybrid model–taking Sun Yat-Sen freeway as an example, IEEE Trans. Veh. Technol. 69 (8) (2020) 8257–8266.
[36] M. Abdollahi, T. Khaleghi, K. Yang, An integrated feature learning approach using deep learning for travel time prediction, Expert Syst. Appl. 139 (2020) 112864.
[37] A. Abdi, C. Amrit, A review of travel and arrival-time prediction methods on road networks: classification, challenges and opportunities, Peer. J. Comput. Sci. 7 (2021) e689.
[38] B.A. Kumar, S. Mothukuri, L. Vanajakshi, Numerical stability of conservation equation for bus travel time prediction using automatic vehicle location data, Int. J. Intell. Transp. Syst. Res. 19 (2021) 141–154.
[39] N. Chiabaut, R. Faitout, Traffic congestion and travel time prediction based on historical congestion maps and identification of consensual days, Transport. Res. Part C: Emerg. Technol. 124 (2021) 102920.
[40] H. Wang, et al., A simple baseline for travel time estimation using large-scale trip data, ACM Transact. Intellig. Syst. Technol. (TIST) 10 (2) (2019) 1–22.
[41] R. Ghiasi, A. Malekjafarian, Feature subset selection in structural health monitoring data using an advanced binary slime mould algorithm, J. Struct. Integrity and Mainten. 8 (4) (2023) 209–225.
[42] L. Zheng, et al., Feature selection-based approach for urban short-term travel speed prediction, IET Intel. Transport Syst. 12 (6) (2018) 474–484.
[43] De Courcy, C., et al., Investigating the relationship between inclement weather and traffic conditions on the m50 motorway: a case study using the mat analysis tool. 2018.
[44] R. Corbally, L. Yang, A. Malekjafarian, Predicting the duration of motorway incidents using machine learning, European Transport Res. Rev. 16 (1) (2024) 14.
[45] M. Rogers, S. Darcy, in: Traffic-flow impact of toll booths on M50 motorway, Dublin. in Proceedings of the Institution of Civil Engineers-Transport, Thomas Telford Ltd, 2007.
[46] D. Laoide-Kemp, M. O'Mahony, Dealing with latency effects in travel time prediction on motorways, Transport. Eng. 2 (2020) 100009.
[47] S. Mirjalili, et al., Salp swarm algorithm: a bio-inspired optimizer for engineering design problems, Adv. Eng. Software 114 (2017) 163–191.
[48] A. Kandiri, et al., Modified Artificial neural networks and support vector regression to predict lateral pressure exerted by fresh concrete on formwork, Int. J. Concrete Struct. Mater. 16 (1) (2022) 64.
[49] H. Izadgoshasb, et al., Predicting compressive strength of 3D printed mortar in structural members using machine learning, Appl. Sci. 11 (22) (2021) 10826.
[50] Tasdemir, Y., Artificial neural networks for predicting low temperature performances of modified asphalt mixtures. 2009.
[51] M.T. Hagan, M.B. Menhaj, Training feedforward networks with the Marquardt algorithm, IEEE Trans. Neural Networks 5 (6) (1994) 989–993.
[52] S. Forouzandeh, K. Berahmand, M. Rostami, Presentation of a recommender system with ensemble learning and graph embedding: a case on MovieLens, Multimedia Tools Appl. 80 (5) (2021) 7805–7832.
[53] H. Chen, et al., Feature selection for imbalanced data based on neighborhood rough sets, Inf. Sci. 483 (2019) 1–20.
[54] H. Wang, M. Hong, Supervised Hebb rule based feature selection for text classification, Inf. Process. Manag. 56 (1) (2019) 167–191.
[55] X. Tang, Y. Dai, Y. Xiang, Feature selection based on feature interactions with application to text categorization, Expert Syst. Appl. 120 (2019) 207–216.
[56] M. Rostami, et al., Review of swarm intelligence-based feature selection methods, Eng. Appl. Artif. Intell. 100 (2021) 104210.
[57] Bansal, P., Biogeography-based optimization and support vector regression for freeway travel time prediction and feature selection. arXiv preprint arXiv: 2007.15212, 2020.
[58] R.-C. Chen, et al., Selecting critical features for data classification based on machine learning methods, J. Big Data 7 (1) (2020) 52.
[59] M. Labani, et al., A novel multivariate filter method for feature selection in text classification problems, Eng. Appl. Artif. Intell. 70 (2018) 25–37.
[60] U.M. Khaire, R. Dhanalakshmi, Stability of feature selection algorithm: a review, J. King Saud Univ. 34 (4) (2022) 1060–1073.
[61] K. Tang, S. Chen, A.J. Khattak, Personalized travel time estimation for urban road networks: a tensor-based context-aware approach, Expert Syst. Appl. 103 (2018) 118–132.
[62] T. Kawatani, et al., Prediction of bus travel time over intervals between pairs of adjacent bus stops using city bus probe data, Int. J. Intell. Transp. Syst. Res. 19 (2) (2021) 456–467.
[63] Y. Shen, et al., TTPNet: A neural network for travel time prediction based on tensor decomposition and graph embedding, IEEE Trans. Knowl. Data Eng. 34 (9) (2020) 4514–4526.
[64] X. Jia, et al., Combined prediction of short-term travel time of expressway based on CEEMDAN decomposition, IEEE Access 10 (2022) 96873–96885.
[65] Z. Zou, H. Yang, A.-X. Zhu, Estimation of travel time based on ensemble method with multi-modality perspective urban big data, IEEE Access 8 (2020) 24819–24828.
[66] D. Li, et al., Feature selection and model fusion approach for predicting urban macro travel time, Math. Probl. Eng. 2020 (1) (2020) 6897965.
[67] L. Kang, et al., Urban traffic travel time short-term prediction model based on spatio-temporal feature extraction, J. Adv. Transport. 2020 (1) (2020) 3247847.
[68] H. Cui, et al., Spatiotemporal correlation modelling for machine learning-based traffic state predictions: state-of-the-art and beyond, Transport Rev. 43 (4) (2023) 780–804.
[69] J. Ou, et al., Short-term traffic flow forecasting for urban roads using data-driven feature selection strategy and bias-corrected random forests, Transp. Res. Rec. 2645 (1) (2017) 157–167.
[70] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn Res. 3 (2003) 1157–1182. Mar.
[71] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comput. Electr. Eng. 40 (2014) 16–28, 1.
[72] R. Genuer, J.-M. Poggi, C. Tuleau-Malot, Variable selection using random forests, Pattern Recognit. Lett. 31 (2010) 2225–2236, 14.
[73] M.-F. Li, et al., General models for estimating daily global solar radiation for different solar radiation zones in mainland China, Energy Convers. Manage. 70 (2013) 139–148.
[74] K. Hosseini, et al., E-bike to the future: Scalability, emission-saving, and eco-efficiency assessment of shared electric mobility hubs, Transport. Res. Part D: Transport and Environ. 133 (2024) 104275.
[75] [cited 2022 18 Aug]; Available from: https://www.rte.ie/news/business/2022/0818/1316570-m50-toll-income-rose-13-last-year-as-traffic-recovered/.
[76] P.A. Lopez, et al., Microscopic traffic simulation using sumo, in: 2018 21st international conference on intelligent transportation systems (ITSC), IEEE, 2018.
[77] M. Guériau, I. Dusparic, Quantifying the impact of connected and autonomous vehicles on traffic efficiency and safety in mixed traffic, in: 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2020.
[78] A. Kandiri, E.M. Golafshani, A. Behnood, Estimation of the compressive strength of concretes containing ground granulated blast furnace slag using hybridized multi-objective ANN and salp swarm algorithm, Constr. Build. Mater. 248 (2020) 118676.
[79] A. Behnood, E.M. Golafshani, Predicting the compressive strength of silica fume concrete using hybrid artificial neural network with multi-objective grey wolves, J. Cleaner Prod. 202 (2018) 54–64.
[80] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Software 69 (2014) 46–61.
[81] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Software 95 (2016) 51–67.
[82] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, Adv. Neural Inf. Process. Syst. 30 (2017).
[83] Molnar, C., Interpretable machine learning. 2020: Lulu. com.