A system for electric vehicle’s energy-aware routing in a transportation network through real-time prediction of energy consumption

Shatrughan Modi · Jhilik Bhattacharya

Received: 31 July 2021 / Accepted: 27 March 2022 / Published online: 18 April 2022
© The Author(s) 2022

Abstract
To tackle the problem of range anxiety of a driver of an electric vehicle (EV), it is necessary to accurately estimate the power/energy consumption of EVs in real time, so that drivers can get real-time information about the vehicle’s remaining range. In addition, it can be used for energy-aware routing, i.e., the driver can be provided with information that on which route less energy consumption will take place. In this paper, an integrated system has been proposed which can provide reliable and real-time estimate of the energy consumption for an EV. The approach uses Deep Auto-Encoders (DAE), cross-connected using latent space mapping, which consider historical traffic speed to predict the traffic speed at multiple time steps in future. The predicted traffic speed is used to calculate the future vehicle speed. The vehicle speed, acceleration along with wind speed, road elevation, temperature, battery’s SOC, and auxiliary loads are used as input to a multi-channel Convolutional Neural Network (CNN) to predict the energy consumption. The prediction is further fine-tuned using a Bagged Decision Tree (BDT). Unlike other existing techniques, the proposed system can be easily generalized for other vehicles as it is independent of internal vehicle parameters. Comparison with other benchmark techniques shows that the proposed system performs better and has a least mean absolute percentage error of 1.57%.

Keywords Energy-aware routing · Multistep traffic speed prediction · Energy consumption estimation · Driving range anxiety · Electric vehicle

Introduction
Electric Vehicles (EVs) are emerging as alternative to fossil fuel vehicles and automobile industry is finding them as a solution to number of problems like environmental pollution and increasing petrol/diesel prices due to limited supply, etc. Although the popularity of EVs is increasing at a rapid rate, but there are some barriers also for their wide spread adoption. Major factors, which affect the consumer perception about EVs and hence their market penetration, have been analyzed by different researchers [1–5]. From these studies, some of the major factors found are range anxiety due to limited driving range, high purchasing price, long time to recharge the battery, and limited availability of public charging infrastructure. Out of these, the limited driving range increase anxiety in the driver that whether he can reach a particular destination or not with current charge is the major contributor for non-adoption of EVs. People feel more comfortable in purchasing plug-in hybrid vehicles in comparison to battery EVs. Although with improvements in battery technology [6,7], consumers have gained some confidence, but accurate and real-time prediction of energy usage can provide a great boost to consumer’s confidence. Therefore, in this work, the focus is on developing a system that can provide reliable real-time energy consumption prediction for different routes to the destination, so that the driver can choose best possible route.

In the literature, researchers have proposed a number of methodologies to estimate the energy consumption of EVs. These methodologies can be divided into three categories, namely, macroscopic, mesoscopic, and microscopic, based on the level of granularity at which they can be used for different applications. The techniques, which can provide total energy consumption estimate for the whole trip that can include multiple connecting roads, are categorized as macroscopic techniques [8–13]. The mesoscopic techniques...
[14–17] can assign energy consumption as a cost to each road in a road network, and then, optimal driving routes to the destination can be found. To get the second-by-second energy consumption estimates, the microscopic techniques [18–22] can be used as they provide more detailed output. The output from microscopic techniques can be aggregated to assign the energy consumption to each road, and thus, they can be used in replacement to mesoscopic techniques. In addition, further aggregation of energy consumption of connecting roads can be used to calculate the total energy consumption for whole trip, which thus can fulfill the purpose of macroscopic techniques. The second-by-second energy consumption output from microscopic techniques can also be used to guide the EV driver’s with instructions like to maintain a particular speed and to select a particular route, etc., based on remaining charge in battery. Hence, these techniques can help in boosting the confidence in EV drivers. Therefore, in this work, a system has been developed which can provide second-by-second energy consumption prediction and can be used to guide the EV driver for energy aware route selection.

Different techniques have been employed by the researchers to develop the methodologies for estimating energy consumption of EVs. Most of the researchers mainly used simulation models [23–26] or regression-based techniques. The regression-based techniques include techniques using linear [13,20,27], polynomial [8,12,14,15,19] or logarithmic [11] regression. The simulation models have a major drawback that to simulate a vehicle, internal vehicle specific parameters are required and these internal parameters are not freely available from manufacturer. In addition, the simulation models cannot be generalized, because they get calibrated to a particular vehicle. The regression-based techniques work on real-world data, but as real-world data are mostly taken from different sensors, so it contains noise in it. This makes regression models to give erroneous results as the regression-based techniques are noise sensitive [28,29]. Other than simulation or regression-based techniques, some techniques were developed using Neural Network (NN) [9,10], Convolutional Neural Network (CNN) [22], and Neuro Fuzzy [30,31]. Although, these techniques show better performance as compared to other regression-based techniques with noisy data, but are not suitable for real-world application. The NN models proposed in [9,10] are macroscopic models, i.e., they gave single output for total energy consumption at the end of the trip. Hence, using these, real-time guidance to the driver about alternative route selection or the optimal speed to maintain, etc. is not possible. Also, these models considered limited number of factors, i.e., speed, jerk, acceleration, and some parameters related to road. Similarly, the CNN-based energy consumption estimation model developed in [22], considered only three parameters, namely, road elevation/grade, tractive effort, and vehicle’s speed as input. This limits the application of these approaches in real world.

In recent years, some of the researchers have also combined multiple techniques to develop hybrid models for energy consumption prediction of EVs. For instance, a machine learning-based hybrid system using Markov Monte Carlo method and XGBoost [32] was developed for predicting future energy consumption. The developed system considers historical driving and environment data for predicting the energy consumption. An ensemble learning approach [33], combining decision trees, random forest, and k-nearest neighbor, was developed for EVs’ energy consumption prediction. The developed system provide prediction results considering number of factors like average trip speed, trip distance, air conditioning, road gradient, etc. Another hybrid model was proposed in [34] for energy consumption estimation, using Recurrent Neural Network (RNN) with attention mechanism and deep neural network. The model takes traffic information and vehicle state as input for generating the prediction results. A combination of residual neural network and Bi-LSTM (Bidirectional Long Short-Term Memory) was proposed in [35] for energy consumption prediction. Another hybrid system [36] using the combination of Deep CNN, LSTM, Bi-LSTM, and ANN was developed for predicting energy consumption of EVs. Similar to [9,10], the hybrid systems [32,33,36] are macroscopic models and cannot be used for real-time driver guidance. Also, although the hybrid models discussed so far have provided encouraging results, but these can be improved further by taking into consideration number of other factors, because these models have considered only a subset of factors for prediction. For instance, the hybrid system proposed in [34,36] ignored environment-related factors. Similarly, [35] ignored the effect of auxiliary loads like air conditioning or heater usage. Number of different studies [37,38] concluded that energy consumption of an EV gets effected by number of factors, such as traffic, road elevation, auxiliary loads, wind speed, wind direction, environment temperature, and initial battery’s state of charge, etc. Hence, for reliable energy consumption estimation, it is necessary that the influence of all these factors be considered.

To fill the gaps discussed above, a microscopic system has been developed which can provide reliable real-time energy consumption estimates based on different influencing factors. The proposed approach uses Deep Auto-Encoders (DAE), cross-connected using latent feature space mapping to predict the traffic speed at multiple time steps in future. For this, it uses the historical traffic speed from number of neighboring roads. Then, a multi-channel Convolutional Neural Network (CNN) has been used to predict the energy consumption based on calculated vehicle speed from predicted traffic speed, road elevation, wind speed, and battery’s initial state of charge, etc. Then, the estimates given by multi-channel CNN are fine-tuned using a Bagged Decision Tree (BDT) by further reducing the prediction error. As mentioned
previously, one of the main problems is to calibrate a simulation model internal parameters of a vehicle is required. As these parameters are not shared by the manufacturers freely, it makes the calibration process very difficult. The proposed system uses six external parameters, namely, historical traffic information, wind speed, temperature, road elevation, initial battery’s SOC, and auxiliary loads, as input, and provide reliable energy consumption estimates. All these input parameters are easy to collect; for instance, historical traffic information is provided by different government agencies for example traffic information for California can be obtained from web-portal of California Department of Transportation Performance Measurement System (PeMS) [39]. Similarly, elevation data for different roads can be obtained using Geographic Information System (GIS), and wind speed and temperature data can be obtained using openly available weather API’s for example, OpenWeatherMap API’s [40]. Another problem discussed above is the noise sensitivity of regression techniques. It is well-known fact that NN-based techniques can handle the noisy data very effectively, and as the proposed system uses CNN and DAE as its modules, the proposed system can work efficiently even with noisy data. Also, unlike other NN-based approaches [9,10], the proposed system provides real-time energy consumption as output, and hence, it can be used to provide real-time instructions to the driver for alternative routes to the destination with less energy consumption or for maintaining a particular speed, so that optimal energy consumption takes place. Following points highlight the major contributions of the current work:

i. A system has been developed which can provide reliable and real-time energy consumption estimates and hence can also be used for guiding the driver in real time.

ii. The proposed approach provides the results considering the influence of all major factors, namely, vehicle acceleration, traffic, temperature, road elevation, auxiliary loads, wind speed, and battery’s SOC.

iii. The proposed system is easy to generalize to any vehicle, because it works independent of internal vehicle parameters, which manufacturers do not share publicly.

iv. The multi-channel CNN and cross-connected DAE make the approach work efficiently even with noisy data, which is generally the scenario in real world. Also, it makes the approach effectively learn the non-linear relationship between the input parameters.

Rest of the paper has been divided into number of sections. First, the details about the real-world dataset used for this study have been discussed in the section “Datasets”. In the section “Analysis of factors influencing energy consumption”, various factors that can influence the energy consumption of an EV are analyzed. The architecture of the proposed microscopic system has been discussed in detail in the section “Proposed system”. The details about the implementation are presented in the section “Experimental setup”. The comparison results of proposed system with other state-of-the-art techniques has been provided in the section “Results and discussion”. Then, “Conclusion” concludes the current work.

Datasets

In this study, three different datasets are used. First and second dataset contains traffic data from multiple loop detectors, located on the highways of Bay Area and Los Angeles city of California state. Third dataset contains data generated from the simulated model of Nissan Leaf, provided in FASTSim (Future Automotive Systems Technology Simulator) [41] which is a very popular simulation tool. The following paragraphs discuss the details of these datasets.

Traffic data for city of Bay Area and Los Angeles were obtained from a reliable data source, web-portal of California Department of Transportation Performance Measurement System (PeMS) [39]. It provides real-time traffic data (like traffic speed, traffic flow and occupancy rate, etc.) from number of loop detector sensors. These sensors are mainly located on different lanes on the freeways of California state. Figure 1a shows the location of 660 different sensors on the map of Los Angeles, for which the traffic data were used in this study. 382 sensors, shown with green marker, out of these 660 sensors were selected that cover five freeways, namely, I110, I5, I405, I10, and US101. Two months (from 1st June, 2017 to 31st July, 2017) of traffic data were used for training and testing the proposed system. Similarly, Fig. 1b shows the 325 sensors on the map of Bay Area. These sensors are installed on eight different highways, namely, SR237, I880, US101, I280, SR85, SR17, SR87, and I680. Traffic data of 6 months (from 1st Jan, 2017 to 30th June, 2017) were collected from these sensors of Bay Area and used in this study for training and testing the proposed system. Furthermore, as there is very less traffic in night, so in this study, traffic data only from 7:00 a.m. to 10:00 p.m. have been used. As traffic data are used from two different location, i.e., Los Angeles and Bay Area; henceforth, these datasets are denoted as PEMS-Los and PEMS-Bay, respectively.

A simulated model of Nissan Leaf given in simulation tool named FASTSim was used to obtain a large amount of data for EV’s energy consumption. The simulated model of Nissan Leaf in FASTSim is based on the vehicle-specific parameters that can be found in [22,42,43]. Similar to this, the other simulated models can be used to obtain dataset for other EVs. This depends on the availability of vehicle-specific parameters from manufacturer like internal resistance of battery and efficiency curve of motor, etc. In this study, the dataset was generated by varying number of different fac-
tors like road elevation (from −20 to 20%), environmental
temperature (from −5 °C to 35 °C with interval of 10 °C),
40 different drive cycles (like New European Driving Cycle
(NEDC) and Highway Fuel Economy Test (HWFET) etc.),
initial state of charge of vehicle’s battery (from 30 to 90%
with interval of 20%), 4 different auxiliary load profiles, and
8 different profiles of wind speed. These eight wind speed
profiles correspond to the first eight categories of wind speed
(out of 13) according to the Beaufort scale [44]. Only first
eight categories were used as last five categories do not pro-
vide the preferable conditions for driving. In this dataset, the
data are recorded at a frequency of 10 Hz. A number of small
partition each of 10 s interval were obtained by partition-
ing the dataset, for conducting the experiments. Therefore,
experiments were conducted with approximately 17 lac par-
titions. It is to be noted that one can generate more data by
changing the different factors like environmental tempera-
ture, road elevation, and drive cycles, etc.

Analysis of factors influencing energy consumption

There are multiple factors (like vehicle speed, acceleration,
road elevation/grade, battery’s initial SOC, auxiliary loads,
wind and environmental temperature, etc.) that influence the
energy consumption of EVs. Following subsections analyze
the influence of these factors on energy consumption.

Vehicle-related factors

The factors related to vehicle include vehicle speed, acceler-
a tion, and auxiliary loads. The vehicle speed and acceleration
mainly depend on driver behaviour and traffic conditions. Figure 2 shows the effect of change in speed/acceleration
on the power/energy consumption of an EV. In the figure,
the vehicle speed represents the speed profile of Highway
Fuel Economy Test (HWFET) drive cycle. It can be observed
that power consumption changes with change in acceleration.
Whenever there is decrease in acceleration, the power
consumption also decreases and vice versa. Also, it can be
seen that area under the power curve represents the energy.
The energy has been categorized into energy consumed and
energy regenerated. From the graph, it can be concluded
that whenever there is negative acceleration, like when driver
applies brake, instead of consuming the energy, the EV can
generate the energy (shown in green color) and store it back
to the battery. This makes EVs more efficient than fossil fuel
vehicles in terms of energy consumption in urban areas with
more traffic.

Similar to speed and acceleration, the auxiliaries, such as
AC/heater and lights, etc., can influence the energy con-
sumption during the trip. To analyze the effects of auxiliaries,
comparison of energy consumption for HWFET drive cycle
with three different auxiliary loads is shown in Fig. 3. The
three different auxiliary loads used for comparison are given
in Table 1. It can be seen from the figure and the table
that energy consumption has a linear relation with auxiliary
loads, i.e., if more auxiliaries are used, more energy will get
consumed. For instance, energy consumption increases by
approximately 1 MJ from auxiliary load profile 1 to auxiliary
load profile 3 with same drive cycle of HWFET. Therefore,
a increase of approximately 12–13% in total energy consump-
tion during the trip has happened, which can be quite critical
for deciding whether the destination can be reached or not.
Therefore, it is important to consider the usage of auxiliary
loads along with speed/acceleration for estimating the energy
consumption.

Driving condition-related factors

Another set of factors that influence energy consumption of
EVs fall under the category of driving conditions, which
include road grade, wind speed, and environmental tem-
perature. Figure 4 show the effect of road grade on energy
consumption of EV for the HWFET drive cycle. In the fig-
ure, the energy consumptions results for four different road
grade profiles are shown by keeping drive cycle, i.e., HWFET
and other factors like auxiliary loads of none, environmen-
tal temperature of 25 °C as constant. It can be observed that
even though the drive cycle and other factors are same, still
the energy consumption is quite different for different road
grades. From the figure, it can be observed that whenever
the road grade is negative (declining road), the energy con-
sumption starts decreasing. This happens due to regenerative
mechanism where energy is regenerated and stored back to
the battery. Due to this regenerative mechanism, it can be
observed that instead of consuming the energy for the trip,
approximately −3 MJ of energy has been stored back to
the battery in Fig. 4c. Similarly, Fig. 4a show that the total
energy consumed with grade profile 4 is approximately 3.5
MJ less than energy consumed with grade profile 1. This
excess energy can further be used for covering more distance
or operating the auxiliaries.

Wind speed can also influence the energy consumption
of an EV during a trip. To analyze the extent of its influ-
ence, energy consumption for HWFET drive cycle at 0%
road grade, 25 °C temperature and no auxiliaries has been
compared with four different wind speeds in Fig. 5. Accord-
ing to Beaufort scale, the four different wind speeds used
for comparison are Calm (<0.5 m/s), Light Breeze (from
1.6 to 3.3 m/s), Moderate Breeze (from 5.5 to 7.9 m/s), and
Strong Breeze (from 10.8 to 13.8 m/s). Here, it is to be noted
that in this study, direction of wind is assumed to be oppo-
site to the vehicle’s travelling direction. From the figure, it
can be observed that the wind speed has linear relation with
Fig. 1 Location of loop detectors on the highways of Los Angeles and Bay area of California state

(a) Loop detectors on the highways of Los Angeles  (b) Loop detectors on the highways of Bay Area

Fig. 2 Relationship of vehicle speed/acceleration and power/energy consumption for HWFET drive cycle. Legend: Vehicle speed (---), Power (----), Energy consumption (---), and Energy regeneration (---)

Fig. 3 Effect of auxiliary loads on energy consumption for HWFET drive cycle. Legend: Vehicle Speed (---), Energy consumption with auxiliary load profile 1 (---), profile 2 (---), and profile 3 (---)
Table 1 Different profiles of auxiliary loads (W) used for energy consumption comparison

| Profile | Mean | Maximum | Minimum | Standard deviation |
|---------|------|---------|---------|--------------------|
| Profile 1 | 0    | 0       | 0       | 0                  |
| Profile 2 | 914.32 | 1144.70 | 760     | 147.62             |
| Profile 2 | 1071.46 | 1421.17 | 883.53  | 190.03             |

Fig. 4 The effect of four different road grade profiles on energy consumption for HWFET drive cycle. Legend: Energy consumption (---) and Road grade profile (---)

energy consumption, i.e., with increase in wind speed, more energy gets consumed and vice versa. Approximately 5 MJ of increase in energy consumption can be observed with strong breeze than calm wind, in Fig. 5.

Other than road grade and wind speed, environmental temperature is also an important factor that can influence the energy usage of an EV during a trip. The environmental temperature mainly affects the battery’s internal resistance which in turn affects the capability of the battery to supply power to the vehicle. The temperature has inverse relationship with battery’s internal resistance, i.e., the battery’s internal resistance increase with decrease in temperature and vice versa. With increase in internal resistance, the capacity of the battery to supply power to the vehicle reduces. This may cause the vehicle to fail to achieve the desired acceleration or to move forward on road with high road inclination, especially when battery’s SOC and temperature are low. Figure 6 shows the effect of temperature along with initial battery’s SOC on energy consumption during the trip with HWFET drive cycle at 0% road grade, calm breeze, and no auxiliaries. The energy consumption is shown as different shades of gray from black to white, where darker shades represent low energy consumption. From the figure, it can be observed that when temperature and initial battery’s SOC is low like temperature of $-5 ^\circ C$ and 30% of initial SOC, the battery supplies less energy to the vehicle and hence less energy consumption. This does not mean that the vehicle was able to cover the trip with less energy, because in such cases, the vehicle stops even before completing the trip. It can be observed that the energy consumption increases with increase in temperature and battery’s SOC, because the battery’s capability to provide power to the vehicle improves to achieve the required acceleration and complete the trip.

In brief, in the above discussion, the individual influence of different factors like road grade, vehicle speed, vehicle acceleration, environmental temperature, initial battery’s SOC, wind speed, and auxiliary loads has been analyzed by keeping others as constant. In real world, this is not the case, as all these factors effect the energy consumption simultaneously during a trip. This makes it a necessity to take into account the influence of all these factors for accurately estimating the energy consumption for a trip.
Fig. 5 The effect of different wind speed profiles on energy consumption with HWFET drive cycle. Legend: Energy consumption with wind speed of category calm ( ), light breeze ( ), moderate breeze ( ), and strong breeze ( ).

Fig. 6 Effect on energy consumption with different temperatures and initial SOC for HWFET drive cycle

Proposed system

A system using deep learning has been developed for accurately predicting the energy consumption of EVs. The proposed system can be utilized to predict the energy consumption on different routes to a destination, and then, the driver can select the best possible route or some routing algorithm can be used for route selection. As discussed in the previous section, to provide accurate estimates the system takes into consideration, the effect of number of factors like traffic, road elevation, wind speed, environmental temperature, etc. Fig. 7, shows the overall framework of the proposed system. There are mainly seven computational modules in the system. In brief, the data pre-processing module pre-processes the raw traffic data by cleaning, clustering, and formatting the data, and then, the traffic prediction module predicts the future traffic speed. The predicted future traffic speed is then used to predict the vehicle’s future speed profile on a particular route. Then, to predict the power consumption, the power consumption estimation module takes seven inputs of vehicle acceleration, speed and road elevation for a route, etc. The re-sampler module is used to increase or decrease the sampling frequency of different parameters, so that all the parameters have same frequency before fine-tuner module uses them as input. The fine-tuner module takes estimated power consumption along with re-sampled seven input parameters of power consumption estimation module and further improve the estimates for power consumption. At last, the state of charge remaining in the vehicle’s battery is calculated by the state of charge calculator module using estimated power consumption given by fine-tuner module. Following subsections discuss the working of each of these seven modules in detail.

Data pre-processing

This module is responsible for pre-processing the raw traffic data collected from different loop detector sensors. As explained in the section “Datasets”, the sensors provide real-time traffic data like traffic flow, occupancy rate and traffic speed, etc. Historical speed data from number of sensors have been used for experiments, in this work. Following subsections explain in detail the process of pre-processing raw traffic data.

Data cleaning

There can be number of anomalies, like outliers due to noise and missing values due to sensor failures, etc., present in traffic data collected from different loop detector sensors. Therefore, it is necessary to clean the data before using it for experiments. Therefore, in this study, data were cleaned beforehand using different data cleaning methods. For instance, it was found that there are some loop detectors for which there are large amount of missing values in the dataset. Hence, such type of sensors were omitted from the dataset, for which the number of missing values crosses a particular limit. For sensors other than these, linear interpolation was used to fill the missing values.
Fig. 7 Framework of the proposed methodology

Neighboring sensors’ selection

To predict traffic of a road, it is necessary to consider the traffic from its adjacent roads also. This effect is known as the traffic’s spatial dependency. Furthermore, this spatial dependency can vary with respect to the time; for instance, one road can contribute more traffic to a particular road in the morning and in evening other road can contribute more. This is known as temporal dependency. Thus, it is necessary that traffic data are clustered according to both spatial and temporal dependencies. To achieve this, an algorithm (see Algorithm 1) has been developed to find the neighboring sensors of a particular sensor based on existing spatial–temporal dependency.

There are multiple input parameters to the algorithm, namely, (i) a list of all \( n \) sensors \( S \), (ii) a \( \tau \times n \) traffic speed matrix \( T_S \), where \( \tau \) represents the total number of time instants, (iii) a \( n \times n \) distance matrix \( D \), where element \( d_{i,j} \in D \) is the shortest distance through road from sensor \( i \) to \( j \), (iv) time instant \( t_0 \), and (v) target sensor \( p \) for which the algorithm has to identify neighboring sensors. First of all, the sensors that are within a distance of \( \delta \) from the target sensor \( p \) are shortlisted and stored in \( S \). The shortest distance of these selected sensors from \( p \) is also stored in \( D_S \). In the next step, the past traffic speed data for the interval of \( t_0 - \Delta t \) of all the selected sensors in \( S \) are recorded in \( T_S \). Then, traffic similarity is obtained between each sensor \( q \in S \) and target sensor \( p \) based on the past traffic data from \( T_S \). Two metrics namely, absolute mean traffic speed difference and Pearson’s correlation coefficient [45], were used for calculating traffic similarity. Therefore, with these steps, three attributes, namely, distance among sensors \( D_S \), absolute mean traffic speed difference \( \Delta_S \), and Pearson’s correlation coefficient \( C_S \), were obtained for each sensor \( q \in S \) with respect to target sensor \( p \). Then, based on these three attributes, all the sensors in \( S \) are ranked using Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [46], which is a multi-criteria decision-making approach. Equal weight \( \lambda \) was used for all the three attributes, but decision-making criteria \( \xi \) were different for each of them. As the correlation should be maximum, so for correlation \( \xi = +1 \) was used. Similarly, \( \xi = -1 \) is used for absolute mean difference and distance as these two attributes should be minimum, for a particular sensor to rank high. From this ranked list, the top \( m \) sensors were selected and stored in \( S^* \) as neighboring sensors of target sensor \( p \). In the current study, top 10 sensors were selected from the ranked list, i.e., \( m = 10 \) was used.

This algorithm was applied for all 382 sensors at different time instants to select the neighboring sensors. Although the target sensor was taken from selected 382 sensors, but the algorithm can output any sensor from total 660 sensors as neighboring sensor, based on the spatial and temporal traffic dependency.

Formatting the traffic data

After obtaining the neighbor sensors for target sensor \( p \), the historical traffic speed data of these selected neighbor sensors \( S^* \) were formatted into a particular matrix (say, \( X \) of size \( l \times m \times c \)). Here, \( l \), \( m \), and \( c \) represent the number of time instances during time interval \( \Delta t \), number of selected neighbor sensors, and number of channels, respectively. For this study, to consider 5 h of past traffic data, \( l \) was taken as 60 as traffic data from loop detectors are available at an interval of 5 mins in the dataset. As discussed in the previous section, \( m \) was taken as 10. The number of channels \( c \) were taken as 4, where each channel represents the traffic data for different day, as shown in Fig. 8. In the figure, \( d \) represents the current day, so first channel contains traffic data for the current day,
channels, i.e., in first channel traffic data for time instances than days, the time instances are also different for different past weeks, the weekly traffic pattern can be captured. Other
d from past 2 weeks, i.e.,
to the traffic data of same day of the week as current day but
d and second channel contains traffic data of the previous day, i.e., d − 1. Similarly, third and fourth channels correspond to the traffic data of same day of the week as current day but from past 2 weeks, i.e., d − 7 and d − 14. By taking data from past weeks, the weekly traffic pattern can be captured. Other than days, the time instances are also different for different channels, i.e., in first channel traffic data for time instances from t0 − Δt to t0 was considered, but in other channels, the time interval is from t0 − Δt/2 to t0 + Δt/2. This was done to capture the past traffic pattern after time instant t0 for the same day but in past 2 weeks. After formatting the past traffic data into the traffic data matrix X, it was passed to the traffic prediction module as input to predict the future traffic speed vector Y were normalized using Eq. (1)

\[
\hat{z}^j = \frac{z^j - \min(z)}{\max(z) - \min(z)}.
\]  

In the above equation, \(z^j\), \(\hat{z}^j\) represent the traffic data and normalized traffic data of \(j^{th}\) sensor, respectively. \(\max(z)\) and \(\min(z)\) are the maximum and minimum of traffic data of all sensors, respectively.

Traffic prediction module

A deep learning model has been developed using Deep Auto-Encoders (DAEs) for predicting multistep traffic speed. Different researchers have employed DAEs to extract robust features from input and then used features further for regression or classification tasks. For instance, DAEs were used by Vincent et al. [47] for extracting robust features from the corrupted inputs. Similar to this, Masci et al. [48] initialized the convolutional neural network with the hierarchical features extracted from DAEs. Although, different researchers have used DAEs for different problems, but it has proved the capability of DAEs for extracting the robust features. Thus, inspired from this fact, two auto-encoders are employed for predicting multistep traffic speed, in this work. There are mainly two components in an auto-encoder, namely an encoder and a decoder. The encoder takes the input and tries to learn an intermediate representation, also known as latent space representation, from the input. This intermediate representation is further used as the input for decoder that tries to regenerate the input from the intermediate representation. The architecture of the proposed traffic prediction model is shown in Fig. 9. The traffic prediction model used two different DAEs: one (represented as DAE_X) for capturing the historical traffic speed patterns and other (represented as DAE_Y) for capturing future traffic speed patterns. The historical traffic speed X is converted to its corresponding latent representation \(Z^X\) by the encoder of the first DAE, \(E_X\), i.e., \(E_X : X \rightarrow Z^X\). The decoder \(D_X\), on the other hand, used this latent representation to regenerate the historical traffic speed, i.e., \(D_X : Z^X \rightarrow X\). Similar to this, the encoder and decoder of the second DAE work on future traffic speed \(Y\) by first converting it to its latent representation \(Z^Y\) and then back to the input future traffic speed. Following equation represents the extraction of \(i^{th}\) latent feature map \((I_i)\) from input \(I\):

\[
I_i = \sigma(I \ast W^i + b^i).
\]  

In the above equation, \(b_i\), \(W^i\), \(\ast\), and \(\sigma\) represent the bias, weight, convolution operation, and activation function. Similarly, the following equation represent the reconstruction
from latent representation:

\[ \hat{I} = \sigma \left( \int_{i \in L} I_i \ast \hat{W} + c \right), \]

(3)

where \( c, \hat{W}, \) and \( L \) represent bias for input channel, transpose convolution operation, and latent feature maps. Therefore, the two DAEs are used to learn the features of individual domains using corresponding intermediate latent representation. For this, mean square error is used as the loss function for input \( I \) and reconstructed output \( \hat{I} \) in both the DAES, given in following equation:

\[ \mathcal{L}_{\text{DAE}} : f(I, \hat{I}) = ||I - \hat{I}||^2. \]

(4)

The basic idea was to employ both the DAES, i.e., \( \text{DAE}_X : E_X - D_X \) and \( \text{DAE}_Y : E_Y - D_Y \), by cross connecting them, i.e., encoder \( E_X \) connected with decoder \( D_Y \), to predict future traffic speed as output using historical traffic speed as input. The motivation for this approach came from the successful application of this idea of cross-connection for image to image translation [49,50]. The cross-connection helps the model to first learn the critical features of both the domains and then use these features to translate from one domain to other. Hence, in this work for cross connection, Latent Feature Mapping Module LFMM was used. The LFMM acts as a bridge between the encoder and decoder by mapping the latent representations of historical traffic speed \( Z^X \) and future traffic speed \( Z^Y \), i.e., \( FM : Z^X \rightarrow Z^Y \). The two DAES \( \text{DAE}_X \) and \( \text{DAE}_Y \) were trained separately; hence, the latent representations \( Z^X \) and \( Z^Y \) are from different dimensional space. Thus, the main purpose of LFMM is to fill the gap between the two latent spaces through proper mapping. As \( \text{DAE}_X \) was trained using historical traffic speed, so the encoder \( E_X \) already learned to extract critical features from the input. Similarly, the decoder \( D_Y \) of \( \text{DAE}_Y \) already learned to decode the future speed \( Y \) from corresponding latent representation \( Z^Y \). The flow of data for individual DAES is shown with red and green arrows, in Fig. 9. The blue arrows are used to represent the flow of data in cross-connected DAES. The architectural details of both the DAES and LFMM are discussed in following subsections.

**Deep auto-encoders (DAEs)**

As explained earlier, in the proposed architecture of traffic prediction module, there are two separate DAES \( \text{DAE}_X \) and \( \text{DAE}_Y \). The encoder \( E_X \) of \( \text{DAE}_X \) has a linear architecture with three convolutional layers each with Tanh activation and batch normalization layer. Convolutional kernels of size 3 \( \times \) 3 are used in all of the convolutional layers. During convolution in each convolutional layer padding of 1 and stride of 2 is used. It has been proved by the past research that deep learning models can achieve better performance with deeper architectures, but it also raises the problem of vanishing gradient that makes training difficult. Thus, the concept of residual learning was introduced to overcome this problem in [51]. Hence, in the encoder \( E_X \), three residual blocks are used to take advantage of residual learning. There are two convolutional layers each having kernels of size 3 \( \times \) 3 with padding and stride of 1. Each convolutional layer is followed by non-linear activation ReLU and batch normalization in each of the residual blocks. In each residual block, a connection is used to add input of residual block with the batch normalized output of next convolutional layer. Dropout layer is also used to avoid overfitting. The architecture of the decoder \( D_X \) is similar to the encoder \( E_X \); just the order of layers is reversed. In decoder \( D_X \), the transposed convolutional layers are used in place of convolutional layers to reverse the effect of convolution.

The second auto-encoder \( \text{DAE}_Y \) also has a linear and simple architecture. Similar to encoder \( E_X \), the encoder \( E_Y \) also has three convolutional layers. Each convolutional layer is followed by a Tanh and batch normalization layer. For convolution, the kernels of size 3 are used in each convolu-
Fig. 9 Architecture of traffic prediction model

Latent feature mapping module (LFMM)

LFMM is used to cross connect the two DAEs. The main purpose of LFMM is to map the historical traffic speed latent features $Z^X$ with future traffic speed latent features $Z^Y$. A linear architecture containing a convolutional, a flatten, and a fully connected layer is used for this. Kernels of size $3 \times 3$ are used in the convolutional layer along with stride of 1 and padding of 2 each for rows and columns. After training the individual DAEs separately, they were cross-connected using LFMM, and then, the complete network was fine-tuned.

Speed profile generator module

In this module, an algorithm (see Algorithm 2) has been developed to calculate the speed profile of the particular route. The algorithm takes mainly four inputs, namely, (i) $S_r$, which represent the list of $k$ sensors on route $r$, (ii) $D_r$, which represent the sequence of distances between sensors, i.e., distance of second sensor from first, then distance of third from second and so on, (iii) future speed prediction $Y_r$ of size $k \times n$, where $n$ is the number of future time steps, and (iv) $t_0$ which represent the initial time instant. In brief, the algorithm first calculates the time $t_d$ a vehicle takes to cover the distance to the next sensor based on the average speed $v_{avg}$. The average speed is the average of current speed $v_{curr}$ at current sensor and the next step’s future speed on next sensor. Then, based on the travel time, using interpolation, the speed on the next sensor is calculated and that becomes the new current speed. This process is repeated for all the sensors on the route $r$. At last, the algorithm returns a time-series $V_r$ which contains traffic speed at the sensor locations of route $r$ at different time instants if a vehicle starts the journey at time $t_0$.

Power consumption estimation (PCE) module

This module has the responsibility to estimate the power consumption for an EV based on environmental, traffic, and road conditions. This has been done by developing a multi-channel CNN [52], given in Fig. 10. The architecture of the multi-channel CNN is inspired from the architecture of the network used in [53] for classifying different hand gestures based on time-series pose data. In PCE module, the proposed multi-channel CNN takes seven time-series as inputs, namely, speed of the vehicle ($V_r$), acceleration ($A_r$), road elevation ($E_{lr}$), auxiliary loads ($A_{xld}$), initial SOC of battery ($B_{soc}$), wind speed ($W_{sp}$), and environment temperature ($E_{tp}$). It has been assumed that a vehicle normally moves with the speed of the traffic. Therefore, the traffic speed profile provided by speed profile generator module is first interpolated before using as input vehicle speed in PCE module. Vehicle acceleration can be easily calculated from vehicle speed. Road
tial SOC (\(B_{soc}\)) and environmental temperature (\(E_{tp}\)), does not vary significantly within 10 s interval. Due to this, for each partition, these parameters were taken as constant and feature extraction was not done for these parameters. For all other input parameters, separate feature extraction module was used to extract the important features. Each feature extraction module has similar configuration with three convolutional branches and one residual branch.

The residual branch is used in each feature extraction module, because during the training, they make backpropagation of gradient better. Due to this, network easily optimizes and achieves better accuracy [51]. Normally residual branches act like an identity function, but here instead of giving input as the output, three average pooling layers are used. The average pooling layers generate a down-sampled output by calculating the average of the particular region of input data. This also makes the network locally invariant, i.e., same features can be extracted by the network regardless of rotation, shifting, or scaling of input [54]. Therefore, pooling layers extract important features along with reducing the scale of the network. Mainly feature extraction happens using the three convolutional branches with similar configuration, as discussed below.

Each convolutional branch of feature extraction module has three convolutional layers activated by ReLU. First, two convolutional layers are followed by average pooling layer and third convolutional layer is followed by a dropout layer and average pooling layer. Convolutional layers in a single convolutional branch have different number of kernels; for instance, the first convolutional layer uses 8 kernels, whereas the second and third use 4 kernels. Each convolutional branch is different from other, as each convolutional branch has convolutional layers with different kernel size, i.e., first branch has layers with kernel size of 7, second has layers with kernel size of 5, and third uses kernels of size 3. Kernels of different sizes help in extracting features with respect to different time resolutions. Dropout layers have been used in each convolutional branch to avoid overfitting of the network. Padding of \(n\) in each convolutional layer has been used. It can be calculated, using the equation \(n = (kernel\_size - 1)/2\), for each convolutional layer based on the size of the kernels used in layer.

The features extracted by each convolutional and residual branch are combined using concatenation and output of size 13 × 12 is obtained from feature extraction module. The output features extracted by each feature extraction module are concatenated and flattened further by the consecutive layers. The flattened features are further concatenated with two input parameters, i.e., temperature and initial battery’s SOC. Then, the feature vector of length 782 is used as input to two consecutive fully connected layers and final power consumption output of size 10 is obtained for interval of 10 s.

Re-sampler module

The input parameters are down-sampled or up-sampled by this module to synchronize the frequency with power consumption output given by PCE module. It is necessary, because the output from PCE module is of 1 Hz frequency, i.e., 10 readings for interval of 10 s, whereas the input parameters are of different frequency. Two input parameters, namely,
battery’s SOC and environmental temperature, are taken as constant for each interval of 10 s. Therefore, these are upsampled to frequency of PCE module’s output, i.e., 1 Hz frequency. Other input parameters are of 10 Hz frequency; hence, they are down-sampled to 1 Hz frequency.

**Fine-tuner module**

The re-sampled input parameters from previous module and the power consumption output from PCE module are combined and then used as input for this module. The fine-tuner module fine-tunes the power consumption estimate further by decreasing the error based on the input parameters. A number of regression models were trained to find out the best one and it has been observed that the Bagged Decision Tree (BDT) gave better results with least error. Breiman [55] proposed an ensemble learning technique called bagging or bootstrap aggregation in which multiple weak learners are combined to get the improved accuracy. Each weak learner, in this case the decision tree, is separately trained using a subset from training dataset. The subsets from training dataset are selected by dividing the dataset using bootstrap resampling. In bootstrap resampling, the training samples are selected randomly from dataset with replacement, i.e., one sample can occur in multiple subsets. After training each learner separately, their output is combined by taking an average value and final fine-tuned power consumption estimate is obtained.

**State of charge calculator module**

The state of charge calculator module takes the fine-tuned estimate from Fine Tuner module and calculate the battery’s remaining state of charge. Following is the equation, given
Training the modules

"Datasets", was used. First, the traffic prediction module was trained using traffic data collected from multiple loop detectors. Second, the power consumption estimation module and fine-tuner module were trained using the data generated from the simulated model of Nissan Leaf, provided in very popular simulation tool named Future Automotive Systems Technology Simulator (FASTSim) [41]. All of these modules were trained with 70% of the corresponding dataset and then tested with remaining 30%. After that, the modules were integrated as part of the complete system and tested again as a whole system. In this section, the implementation details for training and testing of the proposed system have been discussed.

**Training the modules**

The three main modules of the system, i.e., Traffic Prediction module, Power Consumption Estimation module, and Fine Tuner module, were trained separately. While training the fine tuner and power consumption estimation module, the feedback loop, in which the next interval’s initial SOC is obtained using the remaining SOC from previous interval, was not used. However, while testing the complete system, the feedback loop was used.

As explained earlier, the proposed traffic prediction module has two DAEs. After separate training of the each DAE, they were connected using LFMM. Then, the fine-tuning was performed for cross-connected DAEs. Similar to this, the Power Consumption Estimation module was trained. During training these modules, the different layers like convolutional, transposed convolutional and fully connected layers were initialized with Xavier initialization [57]. The Xavier initialization initializes the layer weights using random numbers from the range of $[-\sqrt{\frac{6}{\text{fanin}+\text{fanout}}} , \sqrt{\frac{6}{\text{fanin}+\text{fanout}}} ]$ with uniform distribution. Here, fanin and fanout are the number of input and output connections with the layer, respectively.

As discussed previously in the section “Proposed system”, there are three main modules in the proposed system. First is the Traffic Prediction module, second is Power Consumption Estimation module, and third is Fine Tuner Module. These modules were trained separately and then integrated as part of the whole system. Therefore, to train these modules data from three different sources, described in the section “Datasets”, was used. First, the traffic prediction module was trained using traffic data collected from multiple loop detectors. Second, the power consumption estimation module and fine-tuner module were trained using the data generated from the simulated model of Nissan Leaf, provided in very popular simulation tool named Future Automotive Systems Technology Simulator (FASTSim) [41]. All of these modules were trained with 70% of the corresponding dataset and then tested with remaining 30%. After that, the modules were integrated as part of the complete system and tested again as a whole system. In this section, the implementation details for training and testing of the proposed system have been discussed.

**Experimental setup**

The three main modules of the system, i.e., Traffic Prediction module, Power Consumption Estimation module, and Fine Tuner module, were trained separately. While training the fine tuner and power consumption estimation module, the feedback loop, in which the next interval’s initial SOC is obtained using the remaining SOC from previous interval, was not used. However, while testing the complete system, the feedback loop was used.

As explained earlier, the proposed traffic prediction module has two DAEs. After separate training of the each DAE, they were connected using LFMM. Then, the fine-tuning was performed for cross-connected DAEs. Similar to this, the Power Consumption Estimation module was trained. During training these modules, the different layers like convolutional, transposed convolutional and fully connected layers were initialized with Xavier initialization [57]. The Xavier initialization initializes the layer weights using random numbers from the range of $[-\sqrt{\frac{6}{\text{fanin}+\text{fanout}}} , \sqrt{\frac{6}{\text{fanin}+\text{fanout}}} ]$ with uniform distribution. Here, fanin and fanout are the number of input and output connections with the layer, respectively.

**Testing the modules**

After training the individual modules, they were integrated and tested as a complete system. For this, directed node-edge graph $G$ and $G’$, as shown in Fig. 11, were created where nodes represent the sensor location on freeways of Los Angeles and Bay Area and directed edges represent the roads connecting these sensors. The direction of edges represent the direction of flow of traffic. The elevation at different sensor locations and distance among them was obtained using free web APIs provided by openroute service [59]. A number of routes were identified between different source and destination from the graph $G$ and $G’$. Environmental temperature and wind speed on different dates and time instances were obtained from free web APIs provided by Meteostat [60]. Then, the system was tested for energy prediction accuracy along these routes at different time instances.
Results and discussion

In this section, the results obtained by comparing the performance of proposed system with the other state-of-art techniques are discussed. First of all, the cross-connected deep auto-encoder model proposed in traffic prediction module was compared with multiple machine learning and deep learning techniques. Then, the models which performed best were selected and integrated with other state-of-the-art energy consumption prediction models, and the final integrated models were compared with the proposed system.

Performance evaluation of traffic prediction module

In this section, the performance of traffic prediction module has been compared with number of machine learning (kNN, SVR, ANN, XGBoost, and LASSO) and deep learning (CNN and LSTM) techniques.

i. **ANN**: An Artificial Neural Network (ANN) [61] with three layers was developed for predicting the traffic speed at different time horizons. Sigmoid function was used as the activation function for each neuron. ANN cannot learn the input patterns across time; hence, it lacks the ability to capture temporal dependency.

ii. **LASSO**: Regularized least-square regression (LASSO) [62] was used for traffic speed prediction. Separate LASSO was trained for prediction at different time horizons. Each LASSO was trained using elastic net optimization with Alpha = 0.75 and tenfold cross-validation.

iii. **SVR**: Another popular machine learning algorithm is Support Vector Regression (SVR) [63], which has been used successfully by different researchers for number of regression tasks. Similar to LASSO, separate SVRs, with Radial Basis Function (RBF) kernel, were trained for traffic prediction at different time horizons.

iv. **kNN**: k-Nearest Neighbor (kNN) [64] uses Euclidean distance to determine the k similar input values, and then, it predicts the future traffic speed based on the weighted sum of these values. For selecting the hyperparameter k, cross-validation was used by changing k from 5 to 20.

v. **XGBoost**: For predicting traffic speed, XGBoost [65] was also developed. It is one of the popular techniques that has already been used by number of researchers for different regression-based tasks. To use it for traffic prediction, the historical traffic speed was reshaped into a vector form before passing as input.

vi. **CNN**: One of the most popular deep learning techniques is Convolutional Neural Network (CNN) [54], that has already proved its performance for various tasks. Traffic data in the form of matrices were used for training a CNN. Due to their deep architecture, CNN can identify complex patterns easily irrespective of their spatial orientation, but they lack the capability to learn temporal patterns.

vii. **LSTM**: To learn temporal patterns, Long Short-Term Memory (LSTM) [66] can perform much better than CNNs. As traffic data have inherent temporal dependency, it is developed for traffic prediction.

To have a fair comparison with aforementioned techniques, the best suitable parameters were selected for implementing these techniques. Last 5 h of traffic data of the target sensor was used for training ANN, kNN, SVR, LASSO, XGBoost, and LSTM to predict traffic for next 1 h. The optimal values of hyperparameters were selected for these models using cross-validation; for instance, the kNN model provides the best performance with value of $k = 17$. Traffic data in same matrix format, as for proposed model, were used for training the CNN model. The CNN model contained three convolutional layers activated using ReLU each followed by a average pooling layer, and then, the output is obtained from a dense layer.

Three evaluation metrics, namely, Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE), are used to compare the performance of the above-mentioned state-of-the-art techniques with the proposed traffic prediction module. Following equation can be used to obtain the values for these metrics:

\[
\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_{i}^{\text{act}} - x_{i}^{\text{pred}})^2},
\]

\[
\text{MAPE} = \left( \frac{1}{N} \sum_{i=1}^{N} \left| \frac{x_{i}^{\text{act}} - x_{i}^{\text{pred}}}{x_{i}^{\text{act}}} \right| \right) \times 100,
\]

\[
\text{MAE} = \frac{1}{N} \sum_{i=1}^{N} |x_{i}^{\text{act}} - x_{i}^{\text{pred}}|.
\]

In the above equations, $N$ represents the total number of samples, and $x_{i}^{\text{act}}$ and $x_{i}^{\text{pred}}$ represent the actual and predicted value, respectively. The first two metrics, i.e., RMSE and MAE, measure the absolute deviation between actual and predicted values, whereas the third metric measures the relative deviation. The comparison results of aforementioned benchmark techniques and the proposed traffic prediction module at five different time horizons for both the datasets PEMS-Los and PEMS-Bay are presented in Table 2. The results presented in the table show the effectiveness of proposed approach in predicting traffic at different time horizons. It can be concluded that the proposed approach has performed better than other techniques for all time horizons with respect to both the datasets. The only exception is the prediction performance at 5-min time horizon for PEMS-
Los dataset. Even though at 5-min prediction time horizon for PEMS-Los, the proposed approach does not have the best performance, but the performance is still comparable to other techniques. In the table, with the increase in prediction horizon the performance of each technique decreases, however, there is very less decline in performance for proposed traffic prediction network as compared to other techniques. It is due to the capability of proposed approach to capture spatio-temporal traffic dependency with the help of proposed neighbor sensor selection algorithm, whereas other techniques lack to effectively capture both spatial and temporal dependency. For instance, ANN, LASSO, SVR, kNN, and XGBoost cannot capture temporal dependency. Although, LSTM can effectively understand temporal dependency, but lacks the capability to understand spatial dependency. Similarly, CNN can effectively learn spatial dependency, but lacks the ability to learn temporal dependency.

In brief, from the above discussion and results presented in Table 2, it can be concluded that the performance of proposed traffic prediction module is better in comparison to other benchmark techniques. Other than the above-mentioned techniques used for comparison, a number of more complex techniques have been proposed by different researchers using the PEMS-Bay dataset. This includes techniques like Spatio-Temporal Graph Convolutional Network (STGCN) [67], Graph Multi-Attention Network (GMAN) [68], Graph Wavenet [69], and Diffusion Convolutional Recurrent Neural Network (DCRNN) [70], etc. According to the results provided by respective authors, the MAE of these techniques for 30-min prediction horizon is 1.81, 1.62, 1.63, and 1.74, respectively, and for 60-min prediction horizon is 2.49, 1.86, 1.95, and 2.07, respectively. As compared to these, from Table 2, it can be seen that the proposed traffic prediction module has very less MAE of 1.49 and 1.75 on PEMS-Bay dataset for 30-min and 60-min prediction horizon, respectively. This proves the better performance of proposed technique as compared to other benchmark techniques. For more details, reader is advised to refer [71].

### Performance evaluation of proposed approach

Multiple routes were identified with different source and destination from the node-edge graph $G$ and $G'$, as explained in the section “Testing the modules”. Data from these routes at different time instances were used to test the performance of the proposed approach. To benchmarks the results, the results of proposed approach are compared with multiple state-of-the-art techniques. For this, as discussed in the previous section, three models, namely, XGBoost, LSTM, and CNN, were selected based on the comparison of their performance with traffic prediction module. Then, these selected models were integrated with three state-of-the-art techniques for energy consumption estimation, presented in [19,20,22], and results were obtained for all the routes selected from graph $G$ and $G'$. These energy consumption prediction techniques were implemented as given below:

i. A model was proposed by Galvin [20] for estimating the power consumption of an electric vehicle. The model consist of the following equation which consider speed and acceleration as the main input parameters for calculating the power consumption:

$$ P = 0.7876V^3 - 18.93V^2 + 479.1V + 1507VA, \quad (9) $$

where $V$, $A$, and $P$ represent the EV’s speed, acceleration, and power consumption, respectively.

ii. Yang et al. [19] proposed a model for power consumption estimation of EVs which consider multiple factors for prediction. The model uses following two equations for estimation. Equation (10) is used for calculating power consumption in normal mode and Eq. (11) is used for estimating power generation in regenerative mode

$$ P = \frac{V}{\eta_e \eta_c} \left( \delta m \frac{dv}{dt} + \frac{\rho C_D A}{2} v^2 + mg(f + i) \right) $$

$$ P = \frac{V}{\eta_e \eta_c} \left( \delta m \frac{dv}{dt} + \frac{\rho C_D A}{2} v^2 + mg(f + i) \right) $$
| Technique | Metric | PEMS-Los | PEMS-Bay |
|-----------|--------|----------|----------|
|           | 5-min  | 15-min   | 30-min   | 45-min   | 60-min   | 5-min  | 15-min   | 30-min   | 45-min   | 60-min   |
| ANN       | RMSE   | 2.58     | 4.90     | 6.92     | 8.14     | 9.03    | 3.61   | 4.46     | 5.90     | 7.53     |          |
|           | MAE    | 1.51     | 2.78     | 3.93     | 4.80     | 5.42    | 1.73   | 2.27     | 2.98     | 3.44     |          |
|           | MAPE   | 3.69     | 7.12     | 10.75    | 13.38    | 15.53   | 4.00   | 5.73     | 7.84     | 9.47     | 10.78    |
| LASSO     | RMSE   | 2.58     | 4.98     | 7.06     | 8.37     | 9.29    | 2.73   | 3.88     | 5.83     | 7.03     | 7.89     |
|           | MAE    | 1.50     | 2.84     | 4.09     | 5.05     | 5.80    | 1.17   | 1.99     | 2.98     | 3.78     | 4.39     |
|           | MAPE   | 3.62     | 7.18     | 10.92    | 13.92    | 16.31   | 2.51   | 4.61     | 7.53     | 9.82     | 11.62    |
| SVR       | RMSE   | 2.50     | 4.81     | 6.87     | 8.16     | 9.08    | 2.71   | 3.88     | 5.83     | 7.03     | 7.89     |
|           | MAE    | 1.45     | 2.64     | 3.68     | 4.46     | 5.05    | 1.53   | 2.08     | 2.72     | 3.20     | 3.55     |
|           | MAPE   | 3.47     | 6.45     | 9.38     | 11.65    | 13.44   | 2.51   | 4.61     | 7.53     | 9.82     | 11.08    |
| kNN       | RMSE   | 3.90     | 5.46     | 7.00     | 8.04     | 8.77    | 3.94   | 4.65     | 5.85     | 6.60     | 7.18     |
|           | MAE    | 2.28     | 3.12     | 4.02     | 4.74     | 5.29    | 1.95   | 2.39     | 2.99     | 3.45     | 3.80     |
|           | MAPE   | 5.99     | 8.33     | 10.75    | 13.38    | 16.31   | 2.51   | 4.61     | 7.53     | 9.82     | 11.62    |
| XGBoost   | RMSE   | 2.13     | 4.11     | 5.89     | 6.97     | 7.33    | 2.67   | 3.27     | 4.43     | 5.15     | 5.74     |
|           | MAE    | 1.27     | 2.39     | 3.44     | 4.19     | 4.75    | 1.35   | 1.77     | 2.36     | 2.81     | 3.16     |
|           | MAPE   | 3.09     | 6.10     | 9.04     | 11.35    | 13.09   | 2.76   | 3.94     | 5.57     | 6.82     | 7.82     |
| LSTM      | RMSE   | 2.48     | 4.38     | 5.85     | 6.73     | 7.55    | 3.19   | 3.26     | 4.02     | 4.60     | 5.35     |
|           | MAE    | 1.52     | 2.62     | 3.43     | 4.36     | 4.55    | 1.58   | 1.82     | 2.11     | 2.40     | 2.78     |
|           | MAPE   | 3.71     | 6.56     | 8.87     | 10.43    | 12.11   | 3.29   | 3.97     | 4.72     | 5.42     | 6.54     |
| CNN       | RMSE   | 2.85     | 4.70     | 6.12     | 6.82     | 7.25    | 2.77   | 3.38     | 4.41     | 4.79     | 5.00     |
|           | MAE    | 1.92     | 2.88     | 3.62     | 4.09     | 4.41    | 1.28   | 1.81     | 2.27     | 2.51     | 2.61     |
|           | MAPE   | 4.77     | 7.47     | 9.88     | 11.37    | 12.37   | 2.77   | 4.30     | 5.67     | 6.34     | 6.65     |
| Proposed approach | RMSE | 2.58 | 3.73 | 4.51 | 4.88 | 5.38 | 2.14 | 2.30 | 2.68 | 2.82 | 3.12 |
|           | MAE    | 1.64     | 2.26     | 2.57     | 2.75     | 3.09    | 1.15   | 1.38     | 1.49     | 1.55     | 1.75     |
|           | MAPE   | 4.43     | 6.03     | 6.81     | 7.32     | 8.44    | 2.42   | 3.02     | 3.22     | 3.36     | 3.97     |

The best values are represented using bold notation.

\[ P_{reg} = k v \eta_v \eta_m \left( \delta m \frac{dv}{dr} + \frac{\rho C_D A}{2} v^2 + m g (f + i) \right) \]

\[ + P_{accessory} \]  

\[ (10) \]

\[ P_{reg} = k v \eta_v \eta_m \left( \delta m \frac{dv}{dr} + \frac{\rho C_D A}{2} v^2 + m g (f + i) \right) \]

\[ + P_{accessory} \]  

\[ (11) \]

In above equations, \( P \) represents the power consumption, \( P_{reg} \) is regenerative power, \( v \) is vehicle speed, \( \eta_v \) is transmission efficiency, \( \delta \) is coefficient based on vehicle’s weight, \( m \) is vehicle’s weight, \( f \) is the coefficient for rolling resistance, \( i \) is road grade, \( \rho \) is density of air, \( C_D \) is coefficient of aerodynamic drag, \( A \) area of vehicle’s front, \( P_{accessory} \) is power used for auxiliary loads, \( k \) is percentage of energy that can be restored from braking, and \( \eta_m \) is the motor’s efficiency. Parameter \( k \) lies in the range of \((0,1)\) and can be defined using the following equation:

\[ k = \begin{cases} 
0.5 \times \frac{v}{5} & v < 5 \text{ m/s} \\
0.3 \times \frac{v}{20} + 0.5 & v \geq 5 \text{ m/s}
\end{cases} \]

\[ (12) \]

The model was implemented for comparison by taking the values of \( A, C_D, \) and \( m \) for Nissan Leaf from \([22,42,43]\). Yang et al. \([19]\) provided the values of other parameters, such as \( \delta, \rho, \eta_v, \eta_m, \eta_e, \) and \( f \).

iii. Modi et al. \([22]\) proposed a CNN model which can predict energy consumption for an EV by considering only three input parameters, namely, road elevation/grade, tractive effort, and vehicle speed. The CNN model has a simple architecture with total seven layers (three convolutional layers, two max pooling, one flatten, and one fully connected layer).

The existing energy consumption prediction models, discussed above, were trained and tested using the data from different routes selected from graph \( G \) and \( G' \). Then, these models were integrated with the three selected traffic prediction models, i.e., XGBoost, CNN, and LSTM. The comparison results of proposed approach and these integrated models have been presented in Table 3. The comparison has been presented using three evaluation metrics, i.e., RMSE, MAE, and MAPE, as defined in Eqs. (8), (6) and (7). These met-
Table 3: Performance comparison among other popular approaches and the proposed approach on PEMS-Los and PEMS-Bay datasets.

| Technique        | PEMS-Los | PEMS-Bay |
|------------------|----------|----------|
|                 | RMSE     | MAE      | MAPE | RMSE     | MAE      | MAPE |
| CNN-Galvin       | 9.25     | 7.68     | 29.57| 2.13     | 1.95     | 15.76 |
| CNN-Yang         | 6.67     | 5.32     | 19.68| 2.67     | 2.22     | 17.48 |
| CNN-Modi         | 1.98     | 0.97     | 3.54 | 0.45     | 0.40     | 3.29  |
| LSTM-Galvin      | 9.04     | 7.62     | 29.86| 2.19     | 2.01     | 16.15 |
| LSTM-Yang        | 6.51     | 5.35     | 20.57| 2.83     | 2.43     | 19.25 |
| LSTM-Modi        | 0.99     | 0.65     | 2.55 | 0.44     | 0.39     | 3.27  |
| XGBoost-Galvin   | 9.19     | 7.72     | 30.06| 2.20     | 2.01     | 16.20 |
| XGBoost-Modi     | 6.33     | 5.24     | 20.16| 2.70     | 2.30     | 18.21 |
| XGBoost-Yang     | 1.53     | 0.85     | 3.09 | 0.45     | 0.37     | 3.19  |
| Proposed approach| **0.47** | **0.32** | **1.60** | **0.27** | **0.20** | **1.57** |

The best values are represented using bold notation.

Fig. 12: Performance comparison for sample routes randomly selected from graph $G$ and $G'$ created using PEMS-Los and PEMS-Bay dataset, respectively (dark color represents less error in energy consumption prediction).
rics are calculated based on energy consumption estimation given by the particular model and actual energy consumption given by FASTSim. In Table 3, the notation XGboost–Galvin represent an integrated model, in which XGBoost has been used for traffic speed prediction, and then, model proposed by Galvin [20] has been used for energy prediction. Similar notation has been used for other integrated models. From the table, it can be concluded that the proposed approach provides results with least RMSE, MAE, and MAPE of 0.47, 0.32, and 1.60, respectively, for PEMS-Los and 0.27, 0.20, and 1.57, respectively, for PEMS-Bay dataset. Also, it can be observed that the integrated models, which use CNN model proposed by Modi et al. [22] for energy consumption estimation, performed better than other integrated models. The main reason for this is that the model given by Galvin [20] does not consider the effect of other factors like wind speed, road elevation and auxiliary loads, etc. Similarly, the model proposed by Yang et al. [19] does not consider the effect of environmental temperature, battery’s SOC, etc. Also, these models lack the ability to effectively represent the complex non-linear relationship between different influencing parameters. The model proposed by Modi et al. [22] takes three inputs, namely, vehicle speed, road elevation, and tractive effort. Here, the tractive effort is calculated from the combined effort the vehicle has to make to overcome the force due to road elevation and backward force due to aerodynamic drag, etc. Also, the CNN can learn the non-linear relationship very accurately. Although, the effect of most of factors have been considered, but the effect of auxiliary loads, environmental temperature, and battery’s SOC was not considered in the CNN model of Modi et al. [22]. It can also be observed that the integrated model LSTM-Modi performed better than CNN-Modi and XGBoost-Modi for PEMS—Los and comparable for PEMS—Bay dataset. This is due to better performance of LSTM than CNN and XGBoost for traffic speed prediction, in most of the cases, i.e., traffic prediction for 15 min, 30 min, and 45 min horizon. Therefore, it can be concluded that the error gets accumulated further when the different traffic prediction models were integrated with different energy consumption estimation models.

Similar observations can be drawn from Fig. 12; the figure shows the performance comparison of proposed system and other integrated systems for 100 randomly selected sample routes each from graph $G$ and $G'$ between different source-destination pairs. The comparison has been performed by calculating Absolute Percentage Energy Deviation (APED) for each route using the following equation:

$$\text{APED} = \left( \left| \frac{E_{\text{act}} - E_{\text{est}}}{E_{\text{act}}} \right| \right) \times 100. \tag{13}$$

In the above equation, $E_{\text{act}}$ and $E_{\text{est}}$ represent the actual and estimated energy consumption for a particular route. From Fig. 12, it can be observed that the proposed system provides better estimates for all the routes from graph $G$ and $G'$ with least APED. Also, for most of the routes, the APED is in the range of 0–5%, shown with black color. This validates the results presented in Table 3, as discussed above.

As a special case, out of 100 randomly selected routes from graph $G$, two possible routes between location A and B are shown in Fig. 13. One route, shown in blue color, is of approximately 7.7 miles and other route, part of which is shown in gray color, is of approximately 8.7 miles. Both the routes have a common portion in starting, and then, they get separated. Henceforth, the smaller route will be represented as $R_{-I}$ and longer route will be represented as $R_{-II}$. The energy prediction comparison of proposed approach and integrated models for route $R_{-I}$ on 13th July, 2017 has been presented in Fig. 14. From the figure, it can be observed that the proposed approach shows the same behaviour as the actual, i.e., the deviation of proposed approach’s estimates from actual is very less. Also, it can be seen that each approach is taking different time for completing the trip from location A to B. This is because the proposed approach and integrated models gave different traffic speed estimates and hence follow different speed profile. Due to this, the energy consumption estimation also varies a lot for different approaches, but the proposed approach performs better as compared to all the integrated models.
The comparison of energy estimation by proposed approach and different integrated models for routes $R - I$ and $R - II$ for two different days, i.e., 13th July, 2017 (Thursday) and 16th July, 2017 (Sunday), has been presented in Fig. 15. From the figure, it can be observed that energy consumption is more on weekend in morning, i.e., 16th July, 2017 (Sunday) at 10 a.m. as compared to weekdays morning/evening and weekend evening. This is due to less traffic on Sunday morning; hence, the driver can drive at higher speed and also the driver uses less braking which results in less energy regeneration which can be restored back to the battery. Hence, more energy consumption takes place from the battery. On comparing energy consumption of both the routes, it can be seen that in most of the cases, energy consumption for route $R - II$ is more than route $R - I$. The obvious reason for that is more distance to be covered in route $R - II$, but in Fig. 15c, the energy consumption for route $R - II$ is less than route $R - I$. Here, the driver can save around 1 MJ of energy, i.e., the saving of around 9%, which can be vital in cases where the SOC of battery is already less. This is just one case, but there can be other such cases where the driver can save more energy. Therefore, in such cases, using the estimates provided by the proposed approach, the driver can
select the route $R - II$ over $R - I$ for saving energy, even though the vehicle has to cover more distance.

### Statistical analysis

To further validate the performance of proposed technique, two statistical tests namely, Multiple Comparison Test (MCT) and Kruskal–Wallis $H$ Test (KWT), have been used. KWT is a rank-based non-parametric test based on one-way ANOVA. This test is mainly used to determine if independent samples from two or more groups are significantly different or not. Unlike KWT, the MCT helps to identify the groups that are significantly, different from one another using one-way ANOVA for comparing multiple groups. To validate the performance, these tests were performed in sequential order. First, KWT was applied on APED of prediction results for different routes of $G$, obtained using proposed and other integrated models. Then, using the rank information obtained from KWT, the second statistical test of MCT was applied.

The test results of KWT and MCT are given in Fig. 16. In Fig. 16a, a Box and Whisker plot has been shown for the results obtained from KWT test. The horizontal lines of boxes represent the 1st, 2nd, and 3rd quartiles. Similarly, there are vertical lines passing through boxes. These vertical lines represent the deviation of data. There are some outliers also that are shown using plus symbols in red color. From the Box and Whisker plot, it can be observed that the mean APED for proposed approach is least as compared to mean APED of other integrated systems. Also, the small size of the box for proposed approach validates that there is very less deviation in APED for different routes of $G$. Figure 16b shows the results of MCT applied on rank information obtained from KWT. In the figure, proposed approach has been highlighted using blue color line. Red color lines are used to represent techniques that are significantly different from proposed approach. To represent techniques that are not significantly different, gray color lines have been used. The figure shows that the proposed approach is significantly different from
Table 4   Statistical results of multiple comparison test

| Comparison       | First technique | Second technique | Lower bound | Upper bound | Meanrank difference | p value  |
|------------------|-----------------|------------------|-------------|-------------|---------------------|----------|
| Proposed system  | CNN-Galvin      |                 | −812.181    | −553.738    | −682.96             | 1.27E-07 |
|                  | CNN-Modi        |                 | −252.581    | 5.861       | −123.36             | 7.60E-02 |
|                  | CNN-Yang        |                 | −586.171    | −327.728    | −456.95             | 1.27E-07 |
|                  | LSTM-Galvin     |                 | −817.951    | −559.508    | −688.73             | 1.27E-07 |
|                  | LSTM-Modi       |                 | −230.631    | 27.811      | −101.41             | 2.78E-01 |
|                  | LSTM-Yang       |                 | −588.111    | −329.668    | −458.89             | 1.27E-07 |
|                  | XGBoost-Galvin  |                 | −820.821    | −562.378    | −691.6              | 1.27E-07 |
|                  | XGBoost-Modi    |                 | −235.371    | 23.071      | −106.15             | 2.18E-01 |
|                  | XGBoost-Yang    |                 | −575.471    | −317.028    | −446.25             | 1.27E-07 |

most of techniques. There is some overlap with techniques CNN-Modi, LSTM-Modi, and XGBoost-Modi. This implies the comparable performance of these techniques and proposed approach. However, as discussed above, the proposed technique has least APED as compared to other techniques, so this implies that proposed technique provide better energy consumption estimates.

Statistical results obtained from MCT are provided in Table 4. The first two columns of the table provide the techniques that are compared. For each pair of techniques, the third-to-sixth columns provide lower bound, upper bound, mean rank difference, and p value, respectively. The MCT test was performed by considering significance level α = 0.05. The lower bound and upper bound provide the boundary values for 95% confidence interval. As in the table for most of the techniques, the p value is less than α, it shows that the proposed technique is different significantly. Hence, null hypothesis for these pairs is rejected. This similar behaviour can be observed using the values of lower and upper bound. If the confidence interval does not contain 0.0, then it can be concluded that the techniques compared are significantly different with respect to the significance level of 0.05. There are three pairs for which p value is greater than α. As discussed previously, the proposed approach provides better estimates than these techniques also with least APED. Hence, from this discussion, it can be said with confidence that the proposed technique is statistically better than other compared techniques.

**Conclusion**

A complete system has been developed which can provide the accurate and real-time energy consumption estimates to the EV drivers. These estimates can be used for energy-aware routing, i.e., the driver can be provided with information that which route he should take to save energy. The proposed system takes into account the influence of traffic, wind, road elevation, temperature, battery’s SOC, and auxiliary loads. The process of implementation, training, and testing the proposed system has been elaborated in detail. To validate the results, the performance of the system has also been compared with other state-of-the-art techniques using two real-world traffic datasets, namely, PEMS-Los and PEMS-Bay. The main conclusions are highlighted in the following points:

i. The proposed integrated system can provide reliable real-time energy consumption estimates and can be used for guiding the EV drivers in real time, which thus can reduce driver’s range anxiety.

ii. No internal vehicle parameters are required from manufacturer to train and use the proposed system. Hence, it can be generalized for other vehicles.

iii. The comparison of proposed system with other benchmark techniques validates that the proposed system performs better than other techniques with least mean absolute percentage error of 1.60% and 1.57% for datasets PEMS-Los and PEMS-Bay, respectively.

iv. The proposed system can be used in real world by deploying it to the vehicular embedded system by converting it to TensorFlow Lite format.

**Declarations**

**Conflict of interest**  On behalf of all authors, the corresponding author states that there is no conflict of interest.

**Open Access**  This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material
in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Zhang H, Song X, Xia T, Yuan M, Fan Z, Shibasaki R, Liang Y (2018) Battery electric vehicles in Japan: human mobile behavior based adoption potential analysis and policy target response. Appl Energy 220(January):527–535
2. Wolff S, Madlener R (2019) Driven by change: commercial drivers’ acceptance and efficiency perceptions of light-duty electric vehicle usage in Germany. Transp Res Part C Emerg Technol 105:262–282
3. Feng W, Figliozzi M (2013) An economic and technological analysis of the key factors affecting the competitiveness of electric commercial vehicles: a case study from the USA market. Transp Res Part C Emerg Technol 26:135–145
4. Carley S, Krause RM, Lane BW, Graham JD (2013) Intent to purchase a plug-in electric vehicle: A survey of early impressions in large US cites. Transp Res Part D Transp Environ 18(1):39–45
5. Hackbarth A, Madlener R (2013) Consumer preferences for alternative fuel vehicles: a discrete choice analysis. Transp Res Part D Transp Environ 25:5–17
6. Manzetti S, Mariassiu F (2015) Electric vehicle battery technologies: from present state to future systems. Renew Sustain Energy Rev 51:1004–1012
7. Cano ZP, Banham D, Ye S, Hintennach A, Lu J, Fowler M, Chen Z (2018) Batteries and fuel cells for emerging electric vehicle markets. Nat Energy 3(4):279–289
8. Wu X, Freese D, Cabrera A, Kitch WA (2015) Electric vehicles’ energy consumption measurement and estimation. Transp Res Part D Transp Environ 34:52–67
9. Diaz Alvarez A, Serradilla Garcia F, Naranjo JE, Anaya JJ, Jimenez F (2014) Modeling the driving behavior of electric vehicles using smartphones and neural networks. IEEE Intell Transp Syst Mag 6(3):44–53
10. Felipe J, Amarillo JC, Naranjo JE, Serradilla F, Diaz A (2015) Energy consumption estimation in electric vehicles considering driving style. In: IEEE conference on intelligent transportation systems, proceedings, ITSC 2015-Octob pp 101–106
11. Liu K, Yamamoto T, Morikawa T (2017) Impact of road gradient on energy consumption of electric vehicles. Transp Res Part D Transp Environ 54:74–81
12. Liu K, Wang J, Yamamoto T, Morikawa T (2018) Exploring the interactive effects of ambient temperature and vehicle auxiliary loads on electric vehicle energy consumption. Appl Energy 227(August):324–331
13. Fetene GM, Kaplan S, Mabit SL, Jensen AF, Prato CG (2017) Harnessing big data for estimating the energy consumption and driving range of electric vehicles. Transp Res Part D Transp Environ 54:1–11
14. Qi X, Wu G, Boriboonsomsin K, Barth MJ (2018) Data-driven decomposition analysis and estimation of link-level electric vehicle energy consumption under real-world traffic conditions. Transp Res Part D Transp Environ 64(August):36–52
15. De Cauwer C, Verbeke W, Coosemans T, Faid S, Van Mierlo J (2017) A data-driven method for energy consumption prediction and energy-efficient routing of electric vehicles in real-world conditions. Energies 10(5):608
16. Grubwinkler S, Brunner T, Lienkamp M (2014) Prediction range, for EVs via crowd-sourcing. In: IEEE vehicle power and propulsion conference (VPPC), vol 2014. IEEE, pp 1–6
17. Yao E, Yang Z, Song Y, Zuo T (2013) Comparison of electric vehicle’s energy consumption factors for different road types. Discret Dyn Nat Soc 2013:1–7
18. Zhang R, Yao E (2015) Electric vehicles’ energy consumption estimation with real driving condition data. Transp Res Part D Transp Environ 41:177–187
19. Yang SC, Li M, Lin Y, Tang TQ (2014) Electric vehicle’s electricity consumption on a road with different slope. Phys A 402:41–48
20. Galvin R (2017) Energy consumption effects of speed and acceleration in electric vehicles: laboratory case studies and implications for drivers and policymakers. Transp Res Part D Transp Environ 53:234–248
21. Fiori C, Ahn K, Rakha HA (2016) Power-based electric vehicle energy consumption model: model development and validation. Appl Energy 168:257–268
22. Modi S, Bhattacharya J, Basak P (2020) Estimation of energy consumption of electric vehicles using deep convolutional neural network to reduce driver’s range anxiety. ISA Trans 98:454–470
23. Genikomssakis KN, Mitrengis G (2017) A computationally efficient simulation model for estimating energy consumption of electric vehicles in the context of route planning applications. Transp Res Part D Transp Environ 50:98–118
24. Halmeho T, Rahkola P, Tammi K, Pippila J, Pellikka AP, Manninen A, Ruotsalainen S (2017) Experimental validation of electric bus powertrain model under city driving cycles. IET Electric Syst Transp 7(1):74–83
25. Yaich M, Hachicha MR, Ghariani M (2016) Modeling and simulation of electric and hybrid vehicles for recreational vehicle. In: 16th international conference on sciences and techniques of automatic control and computer engineering, STA 2015, IEEE, 2016, pp 181–187
26. Gao DW, Mi C, Emadi A (2007) Modeling and simulation of electric and hybrid vehicles. Proc IEEE 95(4):729–745
27. Liu K, Wang J, Yamamoto T, Morikawa T (2016) Modelling the multilevel structure and mixed effects of the factors influencing the energy consumption of electric vehicles. Appl Energy 183:1351–1360
28. Kalapanidou E, Avouris N, Craciun M, Neagu D (2003) Machine learning algorithms: a study on noise sensitivity. In: Proceeding of 1st Balkan conference in informatics, no. OCTOBER, pp 356–365
29. Nettleton DF, Orrilois-Puig A, Fornells A (2010) A study of the effect of different types of noise on the precision of supervised learning techniques. Artif Intell Rev 34(4):275–306
30. Shen WX, Chan CC, Lo EW, Chau KT (2002) Adaptive neuro-fuzzy modeling of battery residual capacity for electric vehicles. IEEE Trans Industr Electron 49(3):677–684
31. Dai H, Guo P, Wei X, Sun Z, Wang J (2015) ANFIS (adaptive neuro-fuzzy inference system) based online SOC (State of Charge) correction considering cell divergence for the EW (electric vehicle) traction batteries. Energy 80:350–360
32. Zhang J, Wang Z, Liu P, Zhang Z (2020) Energy consumption analysis and prediction of electric vehicles based on real-world driving data. Appl Energy 275:115408
33. Ullah I, Liu K, Yamamoto T, Zahid M, Jamal A (2021) Electric vehicle energy consumption prediction using stacked generalization: an ensemble learning approach. Int J Green Energy 18(9):896–909
34. Li S, Zhang H, Tan H, Zhong Z, Jiang Z (2021) An attention-based deep learning framework for travel energy consumption prediction using stacked generalization. Adv Civ Eng 2021:1–10
35. Elmi S, Tan KL (2021) DeepFEC: energy consumption prediction model for power grid load forecasting. Adv Civ Eng 2021:1–10
conference 2021—proceedings of the world wide web conference. WWW 2021, ACM, New York, NY, USA, 2021, pp 1880–1890
36. Moawad A, Garumurthy KM, Verbas O, Li Z, Islam E, Freyermuth V, Rousseau, A (2021) A Deep Learning Approach for Macroscopic Energy Consumption Prediction with Microscopic Quality for Electric Vehicles (nov 2021). arXiv:2111.12861
37. Sweeting WJ, Hutchinson AR, Savage SD (2011) Factors affecting electric vehicle energy consumption. Int J Sustain Eng 4(3):192–201
38. Yi Z, Bauer PH (2017) Effects of environmental factors on electric vehicle energy consumption: A sensitivity analysis. IET Electrical Systems in Transportation 7(1):3–13
39. California Department of Transportation, Caltrans Performance Measurement System (PeMS) – State of California. http://pems.dot.ca.gov/. Last Accessed on: 28th Dec, 2021
40. OpenWeatherMap, Weather API from Open Weather Map. https://openweathermap.org/api. Last Accessed on: 17th May, 2021
41. Brooker A, Gonder J, Wang L, Wood E, Lopp S, Ramroth L (2015) FASTSim: A Model to Estimate Vehicle Efficiency, Cost and Performance, in: SAE 2015 World Congress and Exhibition, 2015. https://doi.org/10.4271/2015-01-0973https://doi.org/10.4271/2015-01-0973
42. U.S. Department of Energy, 2013 Nissan Leaf Advanced Vehicle Testing - Baseline Testing Results. Tech. rep. (2013)
43. Burrell T (2013) Benchmarking State-of-the-Art Technologies. Tech. rep, Oak Ridge National Laboratory
44. Met Office, National Meteorological Library and Archive Fact sheet No. 6 – The Beaufort Scale, Tech. rep. (2007)
45. Lin LI-K (1989) A Concordance Correlation Coefficient to Evaluate Reproducibility. Biometrics 45(1):255
46. Behzadian M, Khademohammadi S, Yazdani M, Ignatius B (2012) A state-of-the-art survey of TOPSIS applications. Expert Syst Appl 39(17):13051–13069
47. Vincent P, Larochelle H, Bengio Y, Manzagol PA (2008) Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning, ICML ’08, Association for Computing Machinery, New York, NY, USA, 2008, pp 1096–1103
48. Maschi J, Meier U, Cireşan D, Schmidhuber J (2011) Stacked convolutional auto-encoders for hierarchical feature extraction, in: T. Honkela, W. Duch, M. Girolami, S. Kaski (Eds.), Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 6794 LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 49–54
49. Yoo J, Eom H, Choi YS (2019) Image-to-image translation and Performance, in: SAE 2015 World Congress and Exhibition, 2015. https://doi.org/10.4271/2015-01-0973
50. Rahman NR, Rahman RM (2020) Improvement of the performance of optical handwritten digit recognition by incorporating cross-domain autoencoder-based image to image translation technique. International Journal of Knowledge Engineering and Soft Data Paradigms 7(1):45
51. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2016-Decem, IEEE, 2016, pp 770–778
52. Modi S, Bhattacharyya J, Basak P (2021) Convolutional neural network-bagged decision tree: a hybrid approach to reduce electric vehicle’s driver’s range anxiety by estimating energy consumption in real-time. Soft Comput 25(3):2399–2416
53. Devineau G, Moutarde F, Xi W, Yang J (2018) Deep learning for hand gesture recognition on skeletal data. Proceedings - 13th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2018 (2018) 106–113
54. Ma X, Dai Z, He Z, Ma J, Wang YYY, Wang, YYY (2017) Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction. Sensors (Switzerland) 17(4) (2017)
55. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140
56. Kularatna N (2014) Rechargeable battery technologies: An electronic engineer’s view point, in: Energy Storage Devices for Electronic Systems: Rechargeable Batteries and Supercapacitors, Elsevier, 2014, pp. 29–61
57. Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks, in: Journal of Machine Learning Research, Vol. 9, 2010, pp. 249–256
58. Meteostat, Meteostat. https://dev.meteostat.net/ Last Accessed on: 17th May, 2021
59. Chen T, Guerstrin C (2016) XGBoost: A scalable tree boosting system, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Vol. 13-17-Aug of KDD ’16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 785–794
60. Gu Y, Lu W, Qin L, Li M, Shao Z (2019) Short-term prediction of lane-level traffic speeds: A fusion deep learning model. Transportation Research Part C: Emerging Technologies 62:21–34
61. Sharma B, Kumar S, Tiwari P, Yadav P, Nezhurina MI (2018) ANN based short-term traffic flow forecasting in undivided two lane high ways. Journal of Big Data 5(1):48
62. Liu D, Yu W, Baldi S (2019) Broad Learning for Optimal Short-Term Traffic Flow Prediction, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Vol. 11554 LNCS, 2019, pp. 232–239
63. Ahn J, Ko E, Kim EY (2016) Highway traffic flow prediction using support vector regression and Bayesian classifier, 2016 International Conference on Big Data and Smart Computing, BigComp 2016 (2016) 239–244
64. Cai P, Wang Y, Lu G, Chen P, Ding C, Sun J (2016) A spatiotemporal correlator k-nearest neighbor model for short-term traffic multistep forecasting. Transportation Research Part C: Emerging Technologies 62:21–34
65. Shin T, Guerstrin C (2016) XGBoost: A scalable tree boosting system, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Vol. 13-17-Aug of KDD ’16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 785–794
66. Yu B, Yin H, Zha Z (2018) Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting, in: IJCAI International Joint Conference on Artificial Intelligence, Vol. 2018-July, International Joint Conferences on Artificial Intelligence Organization, California, 2018, pp. 3634–3640
67. Zheng C, Fan X, Wang C, Qi J (2020) GMAN: A Graph Multi-Attention Network for Traffic Prediction. Proceedings of the AAAI Conference on Artificial Intelligence 34(01):1234–1241
68. Wu Z, Pan S, Long G, Jiang J, Zhang C (2019) Graph wavenet for deep spatial-temporal graph modeling, in: IJCAI International Joint Conference on Artificial Intelligence, Vol. 2019-Aug, International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 1907–1913
69. Li Y, Yu R, Shahabi C, Liu Y Diffusion convolutional recurrent network: Data-driven traffic forecasting, arXiv (Jul 2017).
70. Modi S, Bhattacharyya J, Basak P (2022) Multistep traffic speed prediction: A deep learning based approach using latent space mapping considering spatio-temporal dependencies. Expert Systems with Applications 189:116140
Publisher's Note  Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.