Energy Consumption Estimation of Electric Vehicles using different Navigational Parameters

A Thesis

submitted in partial fulfillment of the requirements for the award of the degree of

Doctor of Philosophy

in

Computer Science and Engineering Department

by

Shatrughan Modi
Reg no: 951403002

Under the guidance of

Dr. Jhilik Bhattacharya
Assistant Professor, CSED

Dr. Prasenjit Basak
Associate Professor, EIED

Thapar Institute of Engineering and Technology,
Patiala-147001, Punjab, India
April, 2021

Candidate Declaration

I **Shatrughan Modi**, hereby certify that the work, which is being presented in the thesis, entitled **Energy Consumption Estimation of Electric Vehicles using different Navigational Parameters**, in partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** and submitted to the institution is an authentic record of my own work carried out during the period **January, 2015** to **April, 2021** under the supervision of **Dr. Jhilik Bhattacharya** and **Dr. Prasenjit Basak**. I have also cited the reference about the text(s) / figure(s) / table(s) from where they have been taken.

The matter presented in this thesis has not been submitted elsewhere for the award of any other degree or diploma from any institution.

Date: 20/4/21

Shatrughan Modi

Shatrughan Modi
Candidate

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Date: 20/4/21

Dr. Jhilik Bhattacharya
Supervisor

Dr. Prasenjit Basak
Supervisor

I dedicate this thesis to my Guru, Shri Guru Granth Sahib Ji Maharaj, for all the strength and spiritual support during the journey of this work

Abstract

The demand of Battery Electric Vehicles (BEVs) is increasing at a very fast rate and the automobile industry is considering them as the future of transport. They have number of advantages such as pollution free environment, lower cost of ownership and comfortable ride etc. but even with all these advantages customers are hesitant to buy BEVs due to driver's range anxiety, scarcity of charging infrastructure and long charging time. The driver's range anxiety is emerging as the major concern for customers. Hence, a system has been developed to accurately predict the energy consumption for a BEV based on the different influencing factors such as traffic, road elevation, auxiliary loads, environmental temperature and wind speed etc. The proposed system contains two main modules, one for energy consumption estimation and the other for traffic speed prediction.

For estimating the energy consumption based on different parameters two solutions have been proposed namely, Basic Energy Estimation (BEE) model and Improved Energy Estimation (IEE) model. The BEE model uses a Convolutional Neural Network (CNN) to predict energy consumption based on three parameters namely, vehicle speed, tractive effort and road elevation. Multiple experiments with different variations are performed to explore the impact of number of layers and input feature descriptors. The IEE model uses a hybrid approach by combining a multi-channel CNN with Bagged Decision Tree for energy estimation. The IEE model provides better performance as it considers a number of other factors (such as auxiliary loads, battery's initial state of charge, wind speed and environmental temperature etc.) also for prediction. Unlike existing techniques, the proposed approaches do not require internal vehicle parameters from manufacturer and can easily learn complex patterns even from noisy data. Comparison of results with existing techniques shows that the developed IEE model provides better estimates with least mean absolute energy deviation of 0.08 ± 0.069.

To predict the traffic speed a deep learning based approach namely, Multistep Traffic Speed Prediction (MTSP), has been developed using both the spatial and temporal dependencies. To consider spatio-temporal dependencies, nearby road sensors at particular instant are selected based on the attributes such as traffic similarity and distance. Two pre-trained deep auto-encoders were cross connected using the concept of latent space mapping and the resultant model was trained using the traffic data from selected nearby sensors as input. The MTSP model was trained using the real world traffic data collected from loop detector sensors installed on different highways in Los Angeles. The traffic data is freely available from web portal of California Department of Transportation Performance Measurement System (PeMS). The effectiveness

of the MTSP model was verified by comparing it with number of machine/deep learning approaches.

The proposed IEE model and MTSP model are integrated to form a system which can provide real-time accurate energy consumption prediction for different routes to a destination based on different traffic, road and environmental conditions. Hence, the system can be used to guide the driver for selecting the best possible route with minimum energy consumption so that he/she can reach the destination with the remaining charge present in the battery. This will greatly improve the confidence of the driver and hence will reduce driver's range anxiety.

Keywords: Battery Electric Vehicles, Range Anxiety, Deep learning, Convolutional Neural Network, Bagged Decision Tree, Traffic Prediction, Energy Consumption Estimation

Acknowledgements

First and foremost, I would like to thank and express my deepest gratitude to my Guru, **Shri Guru Granth Sahib Ji Maharaj**, for all the support in the tough times and he alone made this all possible. I would also like to express my gratitude to my supervisors **Dr. Jhilik Bhattacharya** and **Dr. Prasenjit Basak** for their invaluable advice and encouragement at every step of my PhD program. Without their unfailing support and belief in me, this thesis would not have been possible. Their contribution to this thesis goes well beyond their role as an academic supervisor and includes constant support on a personal level without which this journey may never have been completed. And for this, I am truly grateful.

I would like to express my gratitude to our HOD **Prof. Maninder Singh** for his constant motivation and encouragement. I also wish to thank my research committee members and non-teaching staff of the institute for their help and support. I would like to give special acknowledgement to **Dr. Harish Garg** who helped me at every step and kept me motivated. I would also like to thank my friends and colleagues Dr. Ashish Girdhar and Miss Rajanpreet Kaur for their constant support.

I would also like to express my sincere and deep gratitude to my parents Shri Surinder Pal Modi and Smt. Pushpa Devi and my sister Shivani for their love, encouragement, care and support. Finally, thanks to my wife Keffy Goyal for having faith in me and supporting me at every step, without her support, I could not complete my Ph.D program and finally lots of love to my son Jainish Modi.

Shatrughan Modi

Table of Contents

List of Figures

List of Tables

List of Notations

t_{eff}	Tractive Effort
f_{ad}	Opposing aerodynamic drag force
f_{rr}	Opposing rolling resistance force
f_{hc}	Hill climbing gravitational force
f_{la}	Opposing linear acceleration force
f_{wa}	Inertial force of vehicle's rotating parts
v_{sp}	Vehicle's Speed
r_{el}	Road Elevation
v_{acc}	Vehicle's Acceleration
a_{ld}	Auxiliary Loads
w_{sp}	Wind Speed
b_{soc}	State of charge of battery
e_{temp}	Environmental Temperature
E_{dev}	Percentage energy consumption deviation
MAE_{dev}	Mean Absolute Energy Deviation
SOC_i	Battery's state of charge at i^{th} time instant
E_{cap}	Battery's rated energy capacity
P_{est}	Power consumption estimated by the proposed approach
P_{act}	Actual power consumption as given in dataset
$\overline{P_{est}}$	Mean of estimated power consumption
$\overline{P_{act}}$	Mean of actual power consumption
E_{est}	Energy consumption estimated by the proposed approach
E_{act}	Actual energy consumption as given in dataset
P_{reg}	Regenerative power
η_{te}	Transmission efficiency
η_e	Driving efficiency
m	Mass of vehicle
ρ	Air density
C_D	Aerodynamic drag coefficient
$P_{accessory}$	Power consumed by accessories
η_m	Motor efficiency
k	Percentage of energy restored by the motor during braking
\$	List of all sensors

$\mathbb{D}$	Distance matrix of sensors
$\mathbb{T}_{\mathbb{S}}$	Traffic speed matrix of the sensors
$\mathbb{S}^*$	Selected neighboring sensors
fan_{in}	Number of input connections
fan_{out}	Number of output connections
$\mathbb{S}_r$	List of sensors on route r
$\mathbb{D}_r$	Sequence of distances between sensors on route r
$\mathbb{Y}_r$	Future multi-time steps speed prediction of sensors on route r
$\mathbb{V}_r$	Output traffic speed at the sensors on route r

List of Abbreviations

ADVISOR	ADvanced VehIcle SimulatOR
ANL	Argonne National Laboratory
ANN	Artificial Neural Network
APED	Absolute Percentage Energy Deviation
API	Application Programming Interface
APRF	Advanced Powertrain Research Facility
ARIMA	Auto-Regressive Integrated Moving Average
BDT	Bagged Decision Tree
BEE	Basic Energy Estimation
BEV	Battery Electric Vehicle
CAN	Controller Area Network
CNL	Convolution with Non-Linearity
CNN	Convolutional Neural Network
Corr	Correlation
CTJ	Cruising Track Jerk
D^3	Downloadable Dynamometer Database
DAE	Deep Auto-Encoders
DNN	Deep Neural Network
EBJ	Ending Brake Jerk
ECR	Energy Consumption Rate
FASTSim	Future Automotive Systems Technology Simulator
FCL	Fully Connected Layer
FM	Feature Maps
GAF	Gramian Angular Field
GIS	Geographic Information System
GPS	Global Positioning System
HU	Hidden Units
HWFET	Highway Fuel Economy Test
ICE	Internal Combustion Engine
IEE	Improved Energy Estimation
INR	Indian Rupee
kNN	k-Nearest Neighbor
LFMM	Latent Feature Mapping Module

LSTM	Long-Short Term Memory
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MLR	Multiple Linear Regression
MP	Max Pooling
MPTDC	Mean Prediction Time per Drive Cycle
MSE	Mean Square Error
MTF	Markov Transition Field
MTSP	Multistep Traffic Speed Prediction
NEDC	New European Driving Cycle
NKE	Negative Kinetic Energy
NMI	Normalized Mutual Information
NN-MLR	Neural Network - Multiple Linear Regression
OICA	Organisation Internationale des Constructeurs d'Automobiles
PCA	Principle Component Analysis
PCE	Power Consumption Estimation
PeMS	Performance Measurement System
PKE	Positive Kinetic Energy
PSAT	Powertrain System Analysis Toolkit
PSIM	Powersim
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
SBJ	Starting Brake Jerk
SFTP	Supplemental Federal Test Procedures
SGD	Stochastic Gradient Descent
SMJ	Starting Movement Jerk
SNR	Signal to Noise Ratio
SOC	State of Charge
SOH	State of Health
SUMO	Simulation of Urban Mobility
SVR	Support Vector Regression
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution
UDDS	Urban Dynamometer Driving Schedule
VSP	Vehicle Specific Power
WHO	World Health Organization

Chapter 1

Introduction

The road vehicles have become the necessity of everyday life and changed the way of human life drastically. According to sales statistics [1] given by Organisation Internationale des Constructeurs d'Automobiles (OICA), more than 90 million new vehicles were sold in 2019 only and this number was increasing till 2019. This include vehicles of both types i.e. passenger and commercial vehicles. The population of vehicles worldwide has already crossed the 1 billion mark in 2010, excluding off-road vehicles and heavy construction machines as per estimates provided by WardsAuto [2]. With the increase in number of vehicles mainly Internal Combustion Engine (ICE) vehicles, the amount of air pollution is also increasing. According to different studies [3–6], the ICE vehicles used in transportation sector contributes almost 90% CO, 40% NO_x, 25% CO_2 and 25% $PM_{2.5}$ emissions. This affects the health of people significantly. World Health Organization (WHO) estimated that due to outdoor air pollution 4.2 million people died worldwide prematurely in 2016 alone [7]. Anenberg et al. [8] in their study estimated that 3,85,000 deaths have occurred in 2015 due to $PM_{2.5}$ and ozone concentration, caused by transportation sector alone. This is approximately 11.4% of total global deaths caused, due to $PM_{2.5}$ and ozone concentration. This stimulates the automotive industry to look for vehicles other than ICE vehicles. Hence, Battery Electric Vehicles (BEVs) emerge as the strong candidate which can help the automotive industry to overcome the problem of environmental pollution.

1.1 History and Future of BEVs

The concept of BEVs is not new to the automotive industry. They were more popular than ICE vehicles in early 1900s. The BEVs first came into existence around 1820s. Who invented the first EV is not clear because the credit has been given to several inventors. Ányos Jedlik, a hungarian priest invented a small-scale model car which was powered by an electric motor in 1828 [9]. Robert Anderson from Scotland developed a crude electric carriage in year between 1832 and 1839. The carriage was powered by non-rechargeable primary cells. The first generation of BEVs was not suitable for use in daily life because the electric motor and battery

had poor efficiency. Also, the batteries were non-rechargeable so they could not be used again once discharged. Almost half of a century passed before the development of DC electric motor and rechargeable batteries. The first generation of EVs, suitable for practical use on roads, was developed in 1880s. With mass production of rechargeable batteries, BEVs gained more popularity by the end of 19th century [10]. They remained an attractive mode of transport till 1920s, with maximum production in 1912. The popularity of BEVs started declining due to a number of factors [11]. The improvement in road infrastructure between cities increased the demand of vehicles that can provide a long traveling range. With the discovery of new oil reserves the prices of petrol/gasoline decreased and hence they became more affordable. Also, in 1912 the electric starter was invented by Charles Kettering [12]. This made the use of ICE vehicles more convenient. The ICE vehicles were also able to provide long traveling range compared to BEVs. Furthermore, long recharging time and costly battery packs for BEVs made them less popular among consumers. Due to this, by 1930s the BEVs gradually disappeared from market.

With the development of laptops and mobile phones in 1990s, the battery technology improved drastically. Number of new types of batteries came onto the market including nickel-metal hydride, nickel cadmium and lithium-ion etc. Also, due to limited availability of oil resources and increase demand, the prices of petrol/diesel started rising again. Further, the impact on environment due to pollution forced the automotive industry to rethink about BEVs. Due to this, the BEVs again gained the popularity and are commercially available since late 2000s. Figures 1.1 and 1.2, show the trend of growth of BEVs [13] in world in last ten years i.e. from 2010 to 2019.

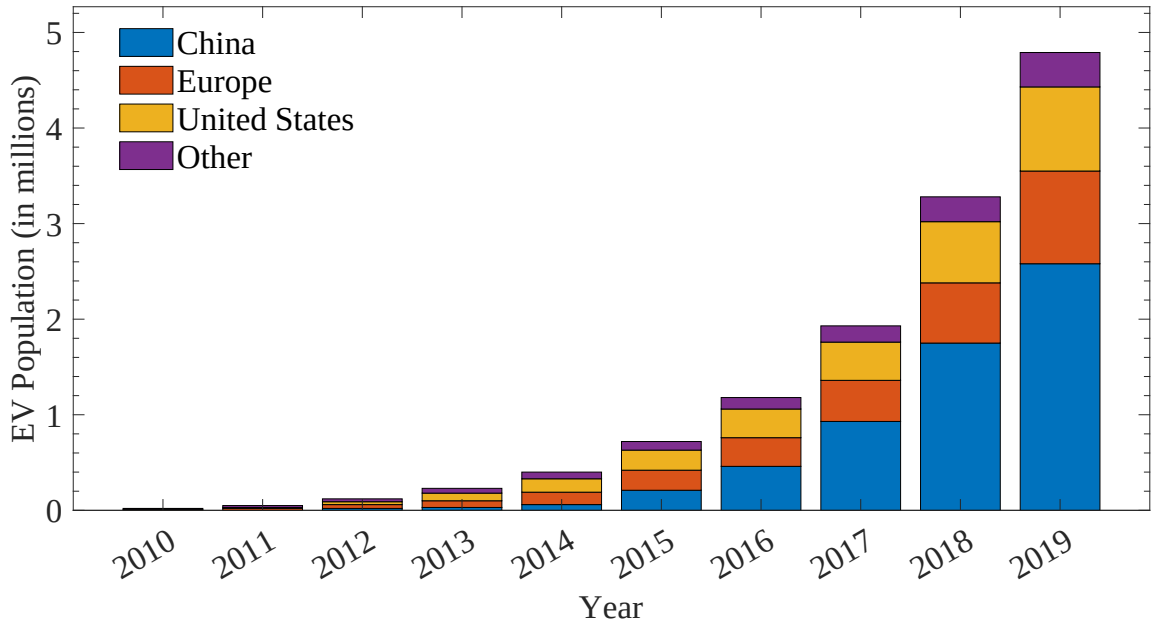


Figure 1.1: Population growth of EVs from year 2010 to 2019

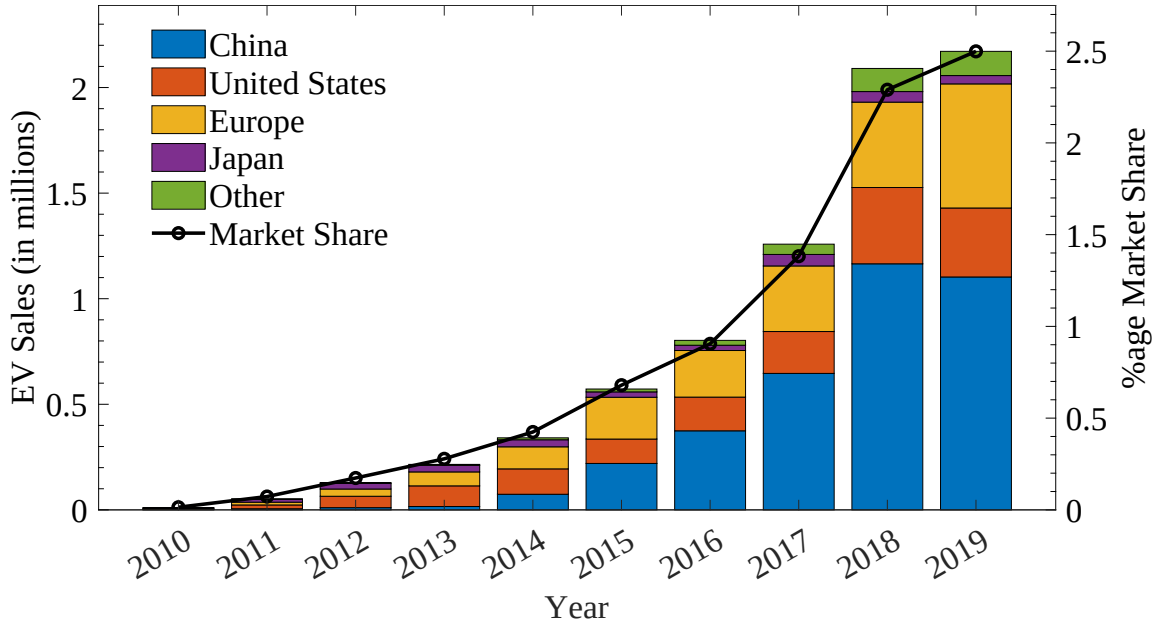


Figure 1.2: EV sales in key country markets and global market share from year 2010 to 2019

From Figure 1.1, it can be concluded that the number of EVs is increasing at a very fast rate. This can be extrapolated and future trends can be found and based on that it has been estimated that by 2030 more than 90 million EVs will be on roads worldwide [13]. Similar trend can be observed in sales of new EVs from Figure 1.2. However, the dramatic slow down in the uptake of sales of EVs in 2019 compared to previous years is worthy to note. From the Figures 1.1 and 1.2, it also can be observed that China is dominating the EV market and this is expected to continue. It is estimated that by 2030 China will hold 49% of global EV market [14]. Among EV manufacturers, Tesla is the leading market player in the global market as of 2019. Tesla sold 370,000 units globally in 2019 which is approximately 16% of the global market share [15].

1.2 Benefits of BEVs

There are number of benefits that BEVs provide in comparison to other vehicles. This section briefly describes all of these benefits of BEVs.

1.2.1 Environment Friendly

One of the major benefit of BEVs is that they are environment friendly. BEVs do not emit harmful pollutants such as carbon monoxide, ozone, hydrocarbons and greenhouse gases etc. However, this benefit may only be local because to recharge the batteries of BEVs the electricity

used come from electric grids where to generate the electricity different fossil fuels are used. This is known as "long tailpipe" of BEVs [16, 17]. The regions where electricity is produced using natural resources such as solar systems or wind turbines, the BEVs can really help in reducing the pollution but in other regions the environmental impact can be not that significant. Thus, it is necessary to study the environmental impact of BEVs using a Life Cycle Assessment (LCA) methodology [18, 19]. This includes raw material extraction, manufacturing of different components, their assembly, usage and maintenance and finally the disposable stage.

Emissions from different sources account for BEVs emission. These sources include emission from manufacturing plants, electricity generation units, waste energy due to vehicle's energy usage efficiency and efficiency of charging process. It has been observed that if coal is used for generating electricity then BEVs are just slightly better than modern ICE vehicles in terms of greenhouse emissions, but BEVs produce more NO_x, PM and SO_2 emissions [20]. However, alternative methods [21, 22] such as solar, wind and hydro etc. can be used to generate electricity at power plants, which can result in almost zero emission and cleaner environment.

1.2.2 Lower Cost of Ownership

Costs of ownership for any vehicle include purchase, running (fuel price) and maintenance (servicing and general wear and tear) expenditures. The purchasing price of BEVs is more than ICE vehicles according to sales figures of 2019 in India [23]. The average cost of the top 10 ICE vehicle models in India is about INR 8 lakh. This is going to increase due to introduction of BS VI emission norms. In comparison to this, the average purchasing price of 3 most accessible BEVs is approximately INR 12.5 lakh. The major part of BEVs purchasing price is for the battery pack. With advancement in battery technology the cost of battery pack is decreasing. This will make BEVs cheaper in the near future. Although, according to sales figures of 2019 in India the upfront cost for BEVs is more than ICE vehicles but the operation/running cost for BEVs is much less than ICE vehicles. Based on recent petrol/diesel and electricity prices, it has been found that the running cost of an ICE vehicle is about INR 3.6-5.6/km, whereas for BEVs this is around INR 1/km. So, BEVs in comparison to ICE vehicles provide more savings with every kilometer driven. According to a study by Sivak and Schoettle [24], in US the average cost to use an ICE vehicle is $1,117/year whereas, for BEV it is $485/year. The maintenance cost of BEVs is also less in comparison to ICE vehicles. Unlike ICE vehicles, the BEVs have lower number of moving parts. The main component in BEVs is the DC motors. The DC motor does not require servicing as frequently as an internal combustion engine. Also, there is much less wear and tear so BEVs do not require oil change and spark plug replacement etc. Due to these factors, the maintenance cost for BEVs reduces to approximately between

half to a third in comparison to the maintenance cost of ICE vehicles. So, due to this the total cost of ownership for BEVs is lower than ICE vehicles and with more usage the user can have much more savings.

1.2.3 Ease of Use

BEVs use electric motors which are mechanically very simple and can be controlled precisely. Due to this, unlike ICE vehicles there is no need of multiple gears in BEVs to match the power curve. Also, electric motors, used in BEVs, provide high torque from rest. So, BEVs can work without gearboxes and torque converters. This helps in improving the driving experience as the driver need not to change the gears again and again in traffic. Also, BEVs provide very smooth and quiet operation i.e. there is no unnecessary noise and vibration during vehicle operation. This makes the user experience more pleasant. The less noise also raises a concern for blind or elder people as there is no usual sound of approaching vehicle [25]. This can be easily resolved by producing warning sounds [26], when the vehicle is moving slowly until the the speed is sufficiently high to generate significant road-tyre interface noise.

1.2.4 High Energy Efficiency

ICE vehicles use petrol/diesel to generate the energy to propel the vehicle whereas BEVs get energy from battery packs. Table 1.1 compares the different energy sources i.e. petrol, diesel and various battery packs in terms of energy density, utilisation and efficiency [27, 28].

Table 1.1: Energy density and efficiency comparison of different sources

	Energy Density (Wh/kg)	Utilisation	Efficiency
Diesel	11800	100%	22%
Petrol	12100	100%	18%
Battery (Lead-Acid)	30-50	80%	80%
Battery (NiCd)	45-80	80%	80%
Battery (NiMH)	60-120	80%	80%
Battery ($LiFePO_4$)	90-160	80%	80%
Battery ($LiCoO_2$)	150-200	80%	80%
Battery ($LiNiCoAlO_2$)	200-260	80%	80%

From the table, it can be observed that petrol/diesel have more energy density in comparison to the available battery technologies. Also, it is recommended to not discharge the batteries to 0%. Normally, it is advised to use the batteries upto 80% of their nominal capacity. This is known as the utilisation factor of batteries. Although, the energy density for batteries is less

and they can not be utilized fully but the energy efficiency of BEVs is very high compared to ICE vehicles. In ICE vehicles most of the energy generated by burning fuel is wasted as heat whereas BEVs can use maximum energy to propel the vehicle. This is mainly because better efficiency of electric motors to convert the energy stored in batteries to move the vehicle. Also, BEVs have a unique feature of regenerative braking i.e. when brakes are applied electric motors of BEVs work as generator and convert the kinetic and gravitational potential energy to electric energy and store this energy back into the batteries. This decreases the energy requirement of BEVs significantly. According to the study carried out by Chłopek et al. [29], BEVs consume approximately 64% less energy compared to ICE vehicles when operated in urban areas due to frequent braking because of high levels of traffic.

1.3 Motivation

Although, BEVs have a number of benefits compared to ICE vehicles their market penetration is still not that high. This is because of a few issues associated with BEVs which act as the barrier for their wide spread adoption. Hübner et al. [30] discussed these issues by conducting a comprehensive survey in UK. A total of 162 electric vehicle drivers were surveyed in UK and a summary of the results are referred in Figure 1.3.

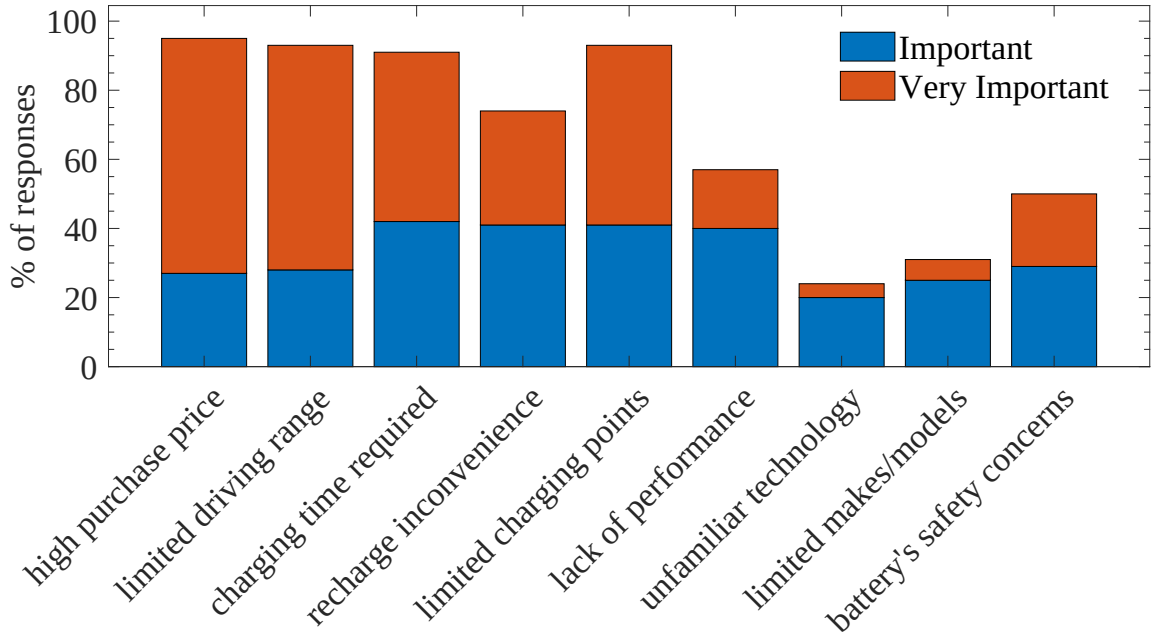


Figure 1.3: Survey responses regarding barriers to the introduction of EVs

Four main barriers towards adoption of BEVs are highlighted in the survey as discussed below:

- High purchasing price
- Limited driving range
- Limited availability of public charging infrastructure
- Time required to recharge the vehicle

Hübner et al. [30] also presented some solutions to these barriers and argued that the research community can play a vital role in removing these barriers. Amongst these, the major challenge faced by the research community sums down to a limited driving range offered by EVs. Zhang et al. [31] also reported that 37.8% EV owners out of 2193 in Japan feels that limited driving range is the major concern for EVs. The driver needs a prior knowledge to confirm whether a particular destination can be reached, given the available battery charge. This is known as the "range anxiety" of the EV driver.

A number of researchers have proposed several solutions to overcome these barriers and improve the adoption rate of BEVs. Most of these are related to overcoming the technical barriers such as improving battery technology to store more energy in the battery [32–36], problem of charging infrastructure [37–39], improving powertrain and electric motor efficiency [40–42]. Large batteries to store more energy can obviously increase the driving range of the BEVs but it also increases the battery weight and cost. Huge amount of investment is required to build fast charging infrastructure. Also, it will take time to have wide spread availability of charging infrastructure. Improving the efficiency of electric motor and using regenerative braking can provide the long driving range, but to increase the driver's confidence there is still need to provide the accurate information to the driver that how far the vehicle can travel with the remaining charge in the battery. For this, it is necessary to accurately predict the energy consumption requirement to reach the destination. As discussed by Ferreira et al. [43], there are number of factors, as shown in Figure 1.4, that can influence the energy consumption of BEVs. A number of researchers have proposed energy consumption estimation algorithms [43–45] by considering only a subset of these factors. So, there is a need for an accurate solution which can provide energy consumption prediction based on all of these influencing factors.

The main aim of this thesis is to increase the driver's confidence by providing real time information to the driver. The driver should be provided with advance information that whether he can reach a particular destination or not given the current charge in the battery. For this, energy consumption has to be predicted accurately based on traffic conditions, route information and weather etc. The driver also should be provided with alternative route information to save some energy. In most of the cases, shortest or fastest route may be the best option, also from

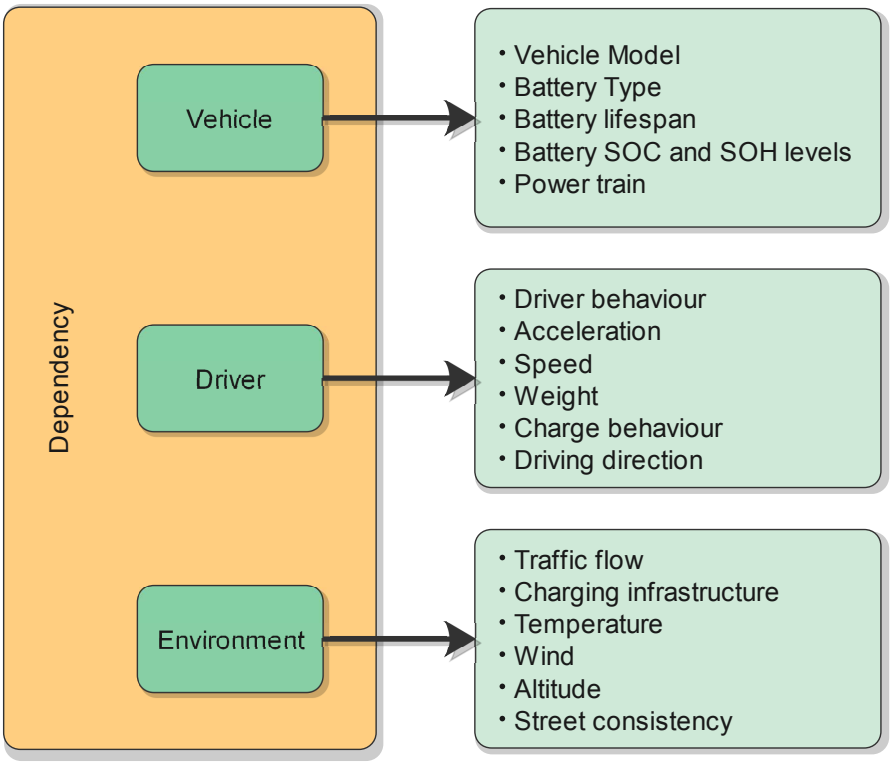


Figure 1.4: Influencing Parameters for Driving Range Prediction

energy consumption point of view. However, there may be some cases where alternative route can provide more energy saving especially the routes with more traffic signalled functions and negative road slope.

1.4 Objectives

The following research objectives have been defined based on the problem discussed above:

1. To classify the terrain based on various surface features.
2. To develop a learning algorithm for estimation of navigational parameters based on energy consumption through simulation/experimentation.
3. To integrate the trained model for augmenting optimal path information for a destination under current conditions.
4. To test and verify the model in different environmental and road conditions.

1.5 Methodology

The following methodology has been adopted to achieve the above defined objectives:

- To achieve the first objective, terrain has been classified based on road elevation/grade. Road elevation, as the main contributor, plays an important role in deciding the energy consumption of an EV because the steeper road inclination the higher the energy consumed from battery. Conversely, if there is negative inclination (i.e. declining road) energy can be generated and stored back to the battery. So, to take it into account, data has been used from roads with varying road grade from -20% (i.e. -11.30°) to 20% (i.e. 11.30°) for training and testing the developed approach.

- The second objective has been achieved by creating two energy consumption estimation models and one traffic prediction model. The first energy consumption estimation model uses Convolutional Neural Network (CNN) to estimate energy consumption based on the tractive effort, vehicle speed and road elevation for a particular trip. The second model is the improved version of the first basic energy model which uses multi-channel CNN and Bagged Decision Tree for estimating energy consumption considering multiple other factors such as wind speed, environment temperature and initial battery's state of charge (SOC) etc. As the energy consumption of an EV also is dependent on the traffic conditions, the traffic prediction model has been developed which uses two deep auto-encoders cross-connected using the concept of latent space mapping.

- To achieve the third and fourth objective, the proposed improved energy estimation model and traffic speed prediction model are integrated to form a system. The integrated system can predict the energy estimation for a route based on traffic conditions, battery's SOC, environment temperature, auxiliary loads, road elevation and wind speed. The integrated system has been verified and tested using the traffic, road and other environmental data from the city of Los Angeles.

1.6 Contributions

The following points highlight the main contribution of the research carried out in this thesis:

1. Two energy consumption estimation models have been developed. One is the Basic En-

ergy Estimation (BEE) model, which uses a Convolutional Neural Network (CNN) for predicting energy consumption. Another, an Improved Energy Estimation (IEE) model has been developed which uses a multi-channel CNN and Bagged Decision Tree (BDT) to give the more accurate energy consumption estimation. Both models also can be used to provide trip level energy consumption estimates.

2. Unlike existing techniques, both the BEE and IEE models take external input parameters and does not require the internal vehicle parameters from manufacturer and hence, can be trained easily for any other vehicle.

3. The BEE model takes only three input parameters, namely, road grade, tractive effort and vehicle speed and can estimate the energy consumption with least Root Mean Square Error(RMSE) of 1.35 KW. The IEE model provides improved estimates with least RMSE of 0.74 KW, as it considers the influence of several other parameters namely, vehicle speed, acceleration, wind speed, auxiliary loads, battery's SOC, environmental temperature and road elevation.

4. The effect of different input features descriptors on the model's performance has also been studied by training the BEE model with three different feature descriptors namely, Covariance, Gramian Angular Field (GAF) and Eigen Vectors. Also, the effect of different number of hidden layers in training the BEE models has been explored.

5. To predict the future traffic speed based on historical traffic data a deep learning based model namely, Multistep Traffic Speed Prediction (MTSP), has been developed which uses two Deep Auto-Encoders (DAE) cross-connected using latent space mapping. This helps to better understand the source and target domain. An algorithm has been developed for neighboring sensor selection to take into account the spatio-temporal dependencies of historical traffic data.

6. The IEE model and MTSP model are integrated to create a system which can provide real-time energy consumption estimation to the driver. The driver also can select the alternative route based on the energy consumption estimates provided by the integrated model.

7. The multichannel CNN and cross-connected DAE makes the integrated system work efficiently even with noisy data, which is generally the case in the real world. Also, it makes the approach effectively learn the non-linear relationship between the input parameters.

1.7 Thesis Outline

The thesis has been organized into 7 chapters. A brief outline of these chapters is given below:

Chapter 1: Introduction This chapter gives an overview of the current status of BEVs in the market. It discusses the history of BEVs along with their future trends that are expected in near future. Also, the various benefits of BEVs such as environmental impact, ease of use and lower cost of ownership etc. are discussed in detail. Problems associated with BEVs, such as driver's range anxiety and limited charging infrastructure etc., that act as a barrier in widespread adoption of BEVs are also discussed. Finally, to overcome the driving range anxiety barrier, objectives of thesis has been formulated.

Chapter 2: Literature Review In this chapter, literature has been discussed for different approaches proposed for estimating the energy consumption based on various influencing parameters. As traffic is an important factor that influences energy consumption of EVs and it is not readily available, approaches developed for predicting the traffic accurately has been discussed also.

Chapter 3: Basic Energy Estimation (BEE) Model In this chapter, a Basic Energy Estimation (BEE) model has been developed. The model uses CNN, a deep learning approach, to estimate the energy consumption by an EV by taking three input parameters, namely vehicle speed, road elevation and tractive effort. These input parameters are time-series and are converted into images for using as input to the proposed model. Three different techniques i.e. Gramian Angular Field, Covariance and Eigen Vectors, are used to convert time-series data into images and to study the effect of different input descriptors on the model's performance. A number of CNN models are trained by varying the number of hidden layers, to study the impact of different number of layers on the model's accuracy. The superiority of the proposed BEE model also has been proved by comparing its performance with several existing techniques.

Chapter 4: Improved Energy Estimation (IEE) Model In this chapter, an Improved Energy Estimation (IEE) model to predict the energy consumption of a BEV has been developed. The model is a hybrid model which uses a multi-channel CNN and Bagged Decision Tree (BDT) for providing the improved energy estimates than BEE model. The model takes seven input parameters, namely vehicle speed, acceleration, wind speed, auxiliary loads, the battery's SOC, environmental temperature and road elevation. The multi-channel CNN extracts individual feature from these input parameters and then combine those

extracted features to obtain an initial energy consumption estimate. The seven input parameters, along with the initial energy estimate provided by multi-channel CNN, are then used by BDT to further fine tune the energy estimates. In order to benchmark the results, the performance of IEE model has been compared with BEE model and other existing techniques.

Chapter 5: Multistep Traffic Speed Prediction (MTSP) Model In orderv to predict the traffic speed at multiple time instants in the future, a Multistep Traffic Speed Prediction (MTSP) model has been developed. The model uses two Deep Auto-Encoders (DAEs) trained separately to learn the critical features of input historical traffic speed and output future traffic speed. After training them separately, they were cross-connected using the concept of latent space mapping. In order to take into consideration the spatial-temporal dependency, the historical traffic data has been taken from neighboring sensors. An algorithm has been developed to select these neighboring sensors based on historical traffic similarity and distance from target sensor. The MTSP model is trained using the real world data collected from several loop detector sensors installed on highways of Los Angeles. The comparison with other existing techniques also has been performed to validate the performance of the MTSP model.

Chapter 6: System Integration In this chapter, the MTSP model, proposed in Chapter 5, and IEE model, proposed in Chapter 4, are integrated as a whole system to provide real-time energy consumption estimates to the driver. The integrated system provides accurate estimates by considering the effect of traffic, road elevation, wind speed, auxiliary loads, initial battery SOC and environmental temperature. The integrated system has been tested on the real-world traffic data collected from highways of Los Angeles. The proposed integrated system has been compared with multiple other integrated systems and it has been found that the proposed integrated system provides estimates with more accuracy.

Chapter 7: Conclusion and Future Work This chapter concludes the thesis by providing the significance and main conclusions drawn from the research work presented in the thesis. Also, some future research direction has been provided for further work in this area.

Chapter 2

Literature Review

In this chapter, the state of art of different techniques developed to estimate energy consumption of BEVs is presented. As discussed in Chapter 1, the energy consumption of BEVs depends on number of factors such as traffic condition, speed of the vehicle, road elevation, wind speed etc. In real life, these factors vary a lot and hence make the estimation of energy consumption a complex problem. Most of these factors can be obtained easily from different sources such as vehicle speed which can be obtained from the vehicle itself, Geographic Information System (GIS) can be used for obtaining road elevation, freely available weather API's (such as API's from OpenWeatherMap [46]) can be used to obtain wind speed and environmental temperature etc. However, future traffic conditions for a particular route can not be obtained easily. So, it is necessary to predict the future traffic so that energy consumption prediction with known statistical confidence can be obtained. For this, a number of researchers have proposed different approaches in literature. So in this chapter, different techniques proposed by researchers for traffic prediction and estimating the energy consumption of BEVs are discussed. The chapter has been divided into three main sections where Section 2.1 discusses the different techniques proposed in literature for energy consumption estimation, Section 2.2 explores the literature for techniques to solve the problem of traffic prediction and Section 2.3 concludes the chapter by summarizing the findings.

2.1 Techniques for Energy Consumption Estimation of BEVs

Energy consumption estimation of BEVs is a complex problem because there are number of factors that can influence the energy use. Over the years, the research community has invested a considerable amount of time and effort to propose a number of models to resolve this problem. These models can be classified into categories from two main perspectives, as shown in Figure 2.1. These perspective are explained in detail as below:

- Modeling methodology: The modeling methodology refers to the type of approach used i.e. rule based and data driven. Rule based models are the "white-box" models that

use the fundamental laws of physics to replicate the interaction and dynamics of various components of BEVs for estimating the energy consumption under different conditions. In comparison to this, the data driven models are the "black-box" models which learn the statistical relationship between inputs and energy output without considering the underlying vehicle dynamics and without understanding the process of energy generation and consumption. These models rely heavily on the input-output data for prediction with known statistical confidence.

- Modeling scale: The modeling scale refers to spatial-temporal resolution of predicted results. There are three main categories (i.e. microscopic, mesoscopic and macroscopic) into which the energy estimation models can be classified based on the modeling scale. The macroscopic models are those which can be used to estimate EVs total energy consumption at the trip level which include orgin and destination along with a number of connecting roads with prevailing traffic conditions. The mesoscopic models are the models used to estimate the EVs energy consumption for each road in the road network so that energy consumption cost can be assigned to each road in the road network and then can be used to plan the optimal route. The microscopic models are the models which provide more detailed second-by-second energy consumption for an EV based on the different influencing factors. The microscopic models can be aggregated to obtain the energy consumption estimates for each link in the road network and hence can serve the purpose of mesoscopic models and then can be further aggregated to estimate the energy consumption for the whole trip and serve the purpose of macroscopic models.

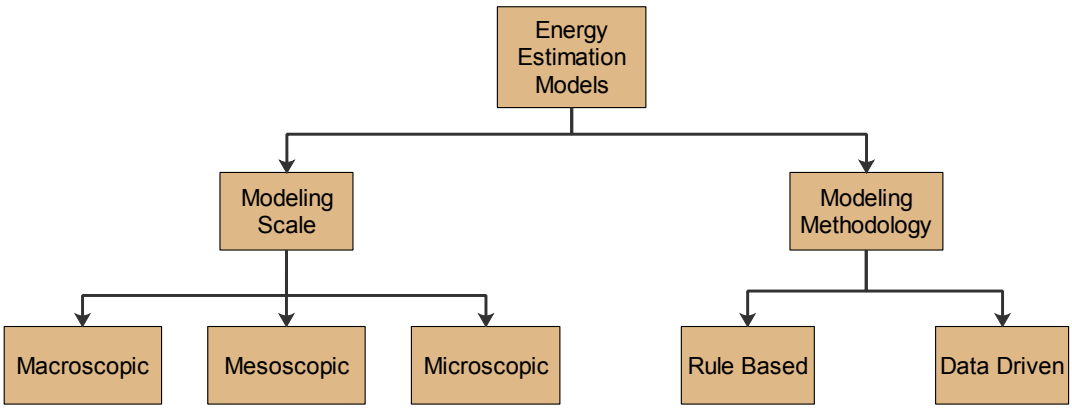


Figure 2.1: Energy consumption estimation models categorized using different perspective

The sections below, discuss the literature for different energy consumption estimation models from the perspective of type of methodology used to develop the model.

2.1.1 Rule Based Models for Energy Consumption Estimation

BEVs have lower number of components and their configuration is less complicated, as compared to ICE vehicles. Due to this, it is relatively easy to represent and understand the flow of energy in BEVs. Also, the energy efficiency of individual components of BEVs is less varied. Therefore, a number of energy consumption models have been proposed by following the rules of physics. Figure 2.2 shows the different forces that act on a vehicle while driving on an inclined road. According to the force balance principle of physics the tractive force t_{eff}, produced by electric motor of EV, should overcome the total opposing force acting on the vehicle which is the sum of the rolling resistance force f_{rr}, the aerodynamic drag f_{ad}, hill climbing force f_{hc} and inertial force due to linear acceleration f_{la}. Based on this, a Vehicle Specific Power (VSP) model, as given by Equation (2.1), has been proposed in [47].

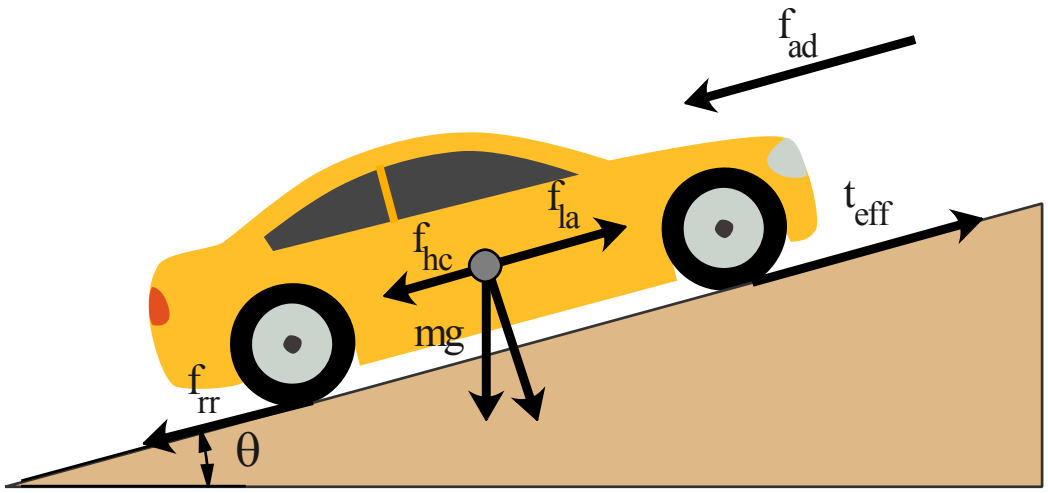


Figure 2.2: Forces acting on the vehicle while driving

$$P/m = VSP = va + gvSin(\theta) + gvf + \frac{1}{2}\rho C_D A v^2 \tag{2.1}$$

In the above equation, P represents the power required from the battery, m is the mass of the vehicle, v is linear velocity, a is linear acceleration, g is gravitational acceleration, θ is the angle of elevation of the road, f is the coefficient of rolling resistance, ρ is air density, C_D is the coefficient of aerodynamic drag and A is the vehicle's frontal area. The VSP model has been used as the basic principle for developing different rule based energy consumption estimation models for EVs which can be further categorised into simulation and physical models.

2.1.1.1 Energy Estimation using Simulation Models

Simulations are models or tools, used to replicate driving behaviour of a vehicle by implementing its different components such as electric motor, battery, transmission mechanism and tires etc. There are various tools (for example Powertrain System Analysis Toolkit (PSAT) [48], ADvanced VehIcle SimulatOR (ADVISOR) [49], Future Automotive Systems Technology Simulator (FASTSim) [50] and Powersim (PSIM) [51] etc.) available that provide the simulation of various vehicle types built in them. Also, they provide the option to create custom tools. Gao et al. [52] discussed the capabilities of different simulation tools for simulating electric and hybrid vehicles. Other than these, tools such as MATLAB/Simulink can be used to implement different vehicles. Different researchers have proposed a number of simulation models of BEVs as discussed below:

Genikomsakis and Mitrentsis [53] developed a simulation model, for energy estimation and route planning, of the Nissan Leaf. The proposed simulation model uses a dynamic approach to mimic the regenerative behaviour of BEV and it also consider the overloading conditions of the electric motor to represent the performance of the BEV on routes with high energy demand. The performance of the simulation models was compared for nine drive cycles using a vehicle simulation tool named, FASTSim. The proposed simulation model was tested for different drive cycles by varying the road grade from -6% to 6%. The model was able to mimic the BEVs behaviour with Mean Absolute Error (MAE) of 45 Wh for cumulative energy consumption.

Four simulation models for a city electric bus were developed by Halmeaho et al. [54]. They also validated the models with the data collected from a bus prototype. The models were different in the type of methods used to simulate the behaviour of the motor. One model used the efficiency map and other models try to represent the losses of the motor as resistive losses. The validation results show that the model using efficiency map of the motor had an error of -3.4% to 5% whereas the models based on resistive loads gave an error of -0.4% to 11.9%.

Donkers et al. [55] studied the effect of driving style, weather, infrastructure and traffic on the performance of EVs. For this, they have developed a model of EV in a simulation tool named, VISSIM. The energy consumption model of the EV is based on the physical model of EV combined with a microscopic traffic model to study the influence of each factor. The energy consumption model was calibrated for BMW i3 vehicle based on the data taken from Argonne National Laboratory [56]. They concluded that factors (such as vehicle speed, acceleration, temperature, air pressure, wind speed, wind direction, road type and traffic intensity etc.) significantly influence the energy consumption of an EV. The driver can achieve high energy efficiency through an economical driving pattern i.e. driving by applying less accelera-

tion.

A simulation model of an EV was developed by Miri et al. [57] based on a real vehicle named, BMW i3. For this, they have used the internal data available for BMW i3 and created the model in MATLAB/Simulink software. The model simulates the behaviour of the vehicle's powertrain system and longitudinal vehicle dynamics. Also, a transmission and a battery model using Thevenin equivalent circuit model has been modeled in the simulation. The developed simulation model also includes the components for simulating regenerative braking and auxiliary loads of an EV. The model was tested using different standard drive cycles and showed satisfactory performance with an error less than 6%.

In general, simulation models use the actual vehicle's internal parameters to replicate the vehicle's behaviour. Hence, they are difficult to generalize as they require calibration according to the specific vehicle and require internal vehicle parameters, including motor efficiency curve, battery internal resistance etc., from the manufacturer. These internal parameters are not readily available and sometimes difficult to obtain which compromises the development of a reliable simulation model.

2.1.1.2 Energy Estimation using Physical Models

Physical models use the laws of physics to understand and replicate the behaviour of actual vehicles. There are number of models developed by different researchers which fall into this category, as discussed below.

Yang et al. [44] proposed an electricity consumption model for an EV based on the different forces acting on a vehicle. The model was tested by varying the road elevation to study its effect on electricity consumption while driving. It has been found that considerable amount of energy gets consumed while travelling uphill whereas, due to regenerative braking significant amount of energy can be regenerated and stored in batteries while driving downhill. For instance, they concluded that for a road distance of $500m$, energy consumption increases from 5.49×10^3 KJ to 7.03×10^3 KJ with increase in road's uphill angle from 1° to 2°.

Wu et al. [58] analysed the data collected for approximately five months from a test vehicle using a data collection system developed using CAN bus data logger and smart phones. Based on the findings of the analysis, they also proposed an analytical model to estimate the energy consumption of BEV. The model basically calculates the tractive effort required by an EV to overcome the different opposing forces such as aerodynamic drag, rolling resistance and road grade resistance etc. The proposed model performed well but the data set used for analysing and developing the model contains data only for one vehicle and one driver which limits its

application.

A fuzzy logic based model has been proposed by Maia et al. [59] for modelling the electricity consumption of EV. Similar to other rule based models, the fuzzy logic based model considers the underlying physical principles of EVs. The model uses fuzzy logic to effectively model the behaviour of regenerative braking considering acceleration, jerk and inclination as the inputs. The results of the proposed model were compared with experiments conducted using a Nissan Leaf driven in urban and suburban regions. The model shows encouraging results but has to be tuned manually which can be quite complex and tedious task.

An instantaneous energy consumption model for Nissan Leaf, considering the speed profile i.e. vehicle speed and acceleration as input, was developed by Fiori et al. [60]. The model was able to capture the regenerative energy efficiency as a function of deceleration of the vehicle. The impact of auxiliary loads on energy consumption was also studied by considering a constant auxiliary load of 850W, 1200W and 2200W keeping the temperature constant at 25°C. It has been found that significant amount of energy is consumed in auxiliary loads. The model was tested on different drive cycles and it has been concluded that energy consumption of EVs is less in urban driving conditions than motorway driving.

A personalized energy consumption model was developed by Jiménez et al. [61] by considering the vehicle dynamics and driving events. Similar to the model developed in [58], the proposed model uses the laws of physics to calculate the tractive force required to overcome different opposing forces. Then the model was extended further to include the effect of different driving events such as left turn, right turn, acceleration and brake. For this, data from different inertial sensors of a smartphone is fed to a neural network which uses some Long-Short Time Memory (LSTM) layers. The neural network classifies the driving events which are then used by the model as driving behaviour input to personalize the estimates.

Luin et al. [62] proposed an energy consumption estimation model for EV based on VSP model. The proposed model also takes into consideration the flow of energy on both sides i.e. the energy consumed for propelling the vehicle forward and the energy generated due to regenerative braking. They coupled the developed model with a microsimulator tool named, Simulation of Urban MObility (SUMO), to study the model's performance in urban traffic. An energy saving of 21.3% was achieved at constant auxiliary loads while testing the model in urban traffic.

In summary, the physical models use the fundamental principles of physics along with physical characteristics of the vehicles so they are easy to understand. however, they are very sensitive to noise i.e. if the data from sensors contains noise these models can fail to provide the good estimates. This makes the real-world application of these models unrealistic as the sensors in

practical-scenario generate noisy data due to interference from different sources.

2.1.2 Data Driven Models for Energy Consumption Estimation

With the advancement of technology, different sensors, telematics and automotive electronics are used to obtain more and different types of data. Researchers have used this data from different sources to develop a number of different data-driven models for estimating the energy consumption of an EV. These data driven models can be categorized, as shown in Figure 2.3. Most of the data-driven approaches developed so far for EVs energy consumption estimation have used regression based techniques such as multiple linear regression [63–71], multilevel linear regression [72], logarithmic regression [69]. Also, a few clustering [73, 74], Neural Network (NN) [75, 76] and decision tree [77] based models were also developed to provide energy consumption predictions with measurable accuracy. The following paragraphs discuss these different types of data driven models in detail.

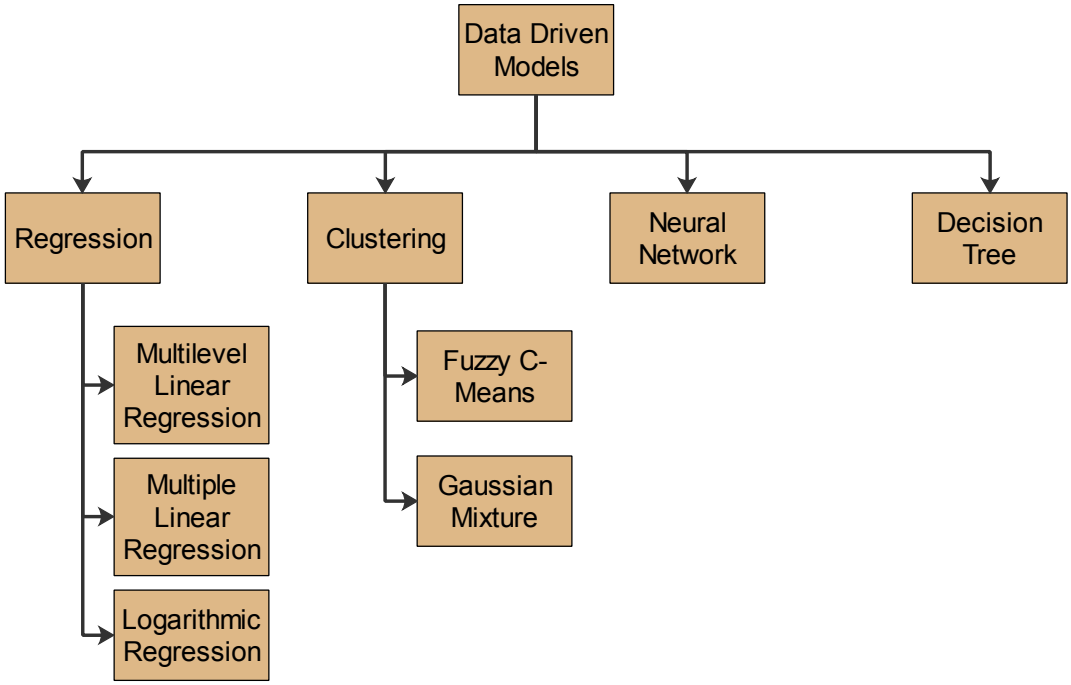


Figure 2.3: Categories of data driven energy consumption estimation models

A multivariate linear regression based energy estimation model for EVs was developed by Galvin [66]. The model takes vehicle speed and acceleration as the main input parameters for predicting the energy consumption. The model was validated using the data from eight commonly used EVs (such as NissanSV, Kia, Mitsubishi and BMW etc.) for a number of different drive cycles. It has been found that the energy efficiency of EVs is compromised significantly when acceleration changes at a high rate regardless of the speed.

Fetene et al. [67] collected and combined data from four different sources namely, GPS driving patterns of 741 drivers over two years, road type, weather conditions and driver characteristics. A linear model to estimate Energy Consumption Rate (ECR) of an EV was proposed and the impact of various factors on ECR was studied. It has been found that approximately 34% more energy is consumed during winters compared to summers. They also concluded that an optimal speed and acceleration can be used to increase the driving range of the vehicle. This is useful to advise the drivers whether they can reach their destination with available charge in battery.

A statistical model using liner regression based on a physical model of an EV was proposed by Yuan et al. [68]. The regression analysis was used to obtain energy consumption characteristics independent of the driving condition. The proposed model considers vehicle's driving speed as the main factor to estimate the energy consumption of an EV under different conditions. However, the model uses test data from a simulation model and therefor lacks practical application as the data set used for the experimentation is very small. Also, it consider only vehicle speed as the main influencing factor, whereas there are number of other factors which influence significantly the energy consumption in an EV such as auxiliary load and environment temperature etc.

A predictive model has been developed for studying the effect of the variation in traffic regulation on energy consumption of EV by López and Fernández [71]. The model predicts the variation in energy consumption of EV with change in traffic parameters that represent the traffic regulations. The stochastic speed profiles for routes were used to minimize the effect of human intervention on energy consumption and then a multiple linear regression model was used to predict the energy consumption based on set of parameters that define the traffic. The proposed model focuses mainly on traffic parameters but there are a number of other influencing factors that need to be considered for obtaining an accurate estimate of energy consumption. These include weather conditions, road conditions and driver behaviour etc.

De Cauwer et al. [63] proposed three models for energy consumption estimation of EV using multiple linear regression based on the equation of vehicle dynamics. The models differ depending on the variables included and the level of aggregation of input parameters. The first model uses the kinematic parameters aggregated for trips as input. The model uses parameters such as travel time, travel distance and temperature. The second model extends the first model by including acceleration data. The third model uses more detailed kinematic data as input to predict energy consumption. The three models show similar performance when compared to the data collected from a Nissan Leaf 2012. They further extended their work in [65] and developed a cascade NN-MLR model for energy consumption prediction. The Neural Network

(NN) in the model was used to predict the speed profile based on road characteristics, weather conditions and traffic and then the third model from [63], developed using Multiple Linear Regression (MLR), was used to predict the energy consumption based on these parameters. The proposed NN-MLR model shows encouraging results as compared to other models.

Qi et al. [70] developed an energy consumption estimation model by analysing the effect of different factors. For this, real world data collected from Nissan Leaf 2013 was used. The data was collected from the vehicle using CAN bus and GPS logger. From the collected data most important features were selected by performing Principle Component Analysis (PCA). It has been found that Positive Kinetic Energy (PKE) and Negative Kinetic Energy (NKE) are the most critical factor that influence the energy consumption in EV. Then, using these features a linear regression model was proposed which show promising results as compared to other regression models.

Liu et al. [69] studied the effect of road elevation/grade using 12 grade ranges and developed eight regression models (4 linear and 4 logarithmic) for energy consumption estimation. For experimentation, data from 492 GPS equipped EVs were collected from February 2011 to January 2013 in Japan. They demonstrated that EVs are more energy efficient than conventional vehicles in areas where road gradient changes frequently, due to regenerative braking. The comparison of proposed eight regression models shows that the logarithmic models performed better than linear models.

Other than these multiple linear and logarithmic regression models, a number of researchers have used multilevel linear regression for developing models to predict energy consumption of EVs. For instance, a number of multilevel mixed-effects linear regression models were proposed by Liu et al. [64] for estimating the energy consumption of EV. The impact of change in driving behaviour under different road and traffic environment was studied by analysing the data collected for 12 months from 68 EVs in Japan. Based on the analysis they proposed three types of multilevel models: one two-level random intercept model, two three-level mixed-effects models and three two-level mixed-effects models. They compared the performance of these models and found that one of the three-level mixed-effects models outperformed the others. Although the models performed well, these models lack the ability to mimic the behaviour of regenerative braking as there was no term defined in the models to consider that.

Further, Liu et al. [72] extended their research by studying the impact of temperature and auxiliary loads. They proposed three models, calibrated using ordinary least square and multilevel mixed-effects linear regression, for estimating energy consumption. From the study, they inferred that the temperature range of $21.8 - 25.2$°C is the most economical in terms of energy efficiency. They have also concluded that with proper usage of auxiliary loads vehicle energy

consumption can be reduced by approximately 9.66% per kilometre which can be vital in scenarios when charge in battery is already low.

Some researchers have used different approaches such as clustering, neural networks and decision trees to develop models for obtaining estimates of energy consumption of an EV. For example, Xu and Wang [73] proposed a power consumption prediction model for EV using clustering. For this, vehicle historical data, elevation of selected route and real-time road congestion data for route were used. In the first step, the energy consumption for different drive cycle categories was obtained from historical vehicle data by applying principle component analysis and Fuzzy C-means clustering algorithm. Next the future vehicle speed profile was predicted using road elevation and real-time congestion data for particular route. Finally, based on future vehicle speed profile the energy consumption was obtained by identifying the drive cycle. The model was verified by conducting 10 real vehicle tests and model was found to perform consistently well for all tests with averagely 4.15% deviation from the actual energy consumption.

Similarly, a cyber-physical system based approach was developed by Lv et al. [74] considering vehicle characteristics and the different driving styles to optimally control the EV for best performance in terms of energy. To recognize different driving styles namely, aggressive, conservative and moderate, data is clustered using an unsupervised machine learning algorithm named Gaussian mixture model. The experimental results confirm that vehicles can perform better with respect to energy consumption in conservative, moderate and aggressive driving style, when an optimized control strategy is used.

Diaz Alvarez et al. [75] trained Artificial Neural Networks (ANN) for estimating the energy consumption of EVs. For this, the neural networks were given input of vehicle speed, acceleration, and jerk. They further classified the acceleration into positive and negative acceleration. Similarly, the jerk was classified into Starting Movement Jerk (SMJ), Cruising Track Jerk (CTJ), Starting Brake Jerk (SBJ) and Ending Brake Jerk (EBJ). The mean and variance of these parameters were used as input to the neural network to obtain energy consumption output for a trip. Felipe et al. [76] extended the work of [75] by adding route information to the input of the neural network. The route information contains information such as road grade, number of lanes etc. They trained the neural network with 137 inputs and 1 output of total energy consumed during the trip. The ANNs developed in [75, 76], gave only one output of total energy consumed for the trip at the end of the trip. So, these models cannot be used in real time to guide the driver about remaining energy in the battery.

Zhang et al. [77] developed an energy consumption prediction algorithm for EVs based on real-world driving data collected from 55 electric taxis in Beijing. They analysed different

factors related to vehicle (velocity, regenerative braking and acceleration), environment (traffic and temperature) and driver behaviour that effect the energy consumption of EV from the collected data. Based on the analysis, they developed a machine learning model using decision trees (XGBoost) to predict the energy consumption of an EV under different conditions. The proposed approach shows considerable improvement in terms of performance when compared to traditional linear regression models.

The approaches discussed above mainly used regression techniques, specifically multiple linear regression. These regression techniques rely heavily on the availability of real-world data and vary in the extent to which they can be linked to underlying physical principles. Also, these techniques are sensitive to noise [78]. As real-world data is obtained from different sensors which generally provide noisy data due to interference from number of different sources, these techniques lack applicability in real-world. Although, the techniques based on NN [75, 76] can handle the noisy data better than other regression techniques, they also cannot be used in real-world. The NN based models provide single energy consumption output for the whole trip. Hence, these models cannot be used by the drivers for real-time guidance based on the current energy consumption to maintain an optimal speed or to take an alternative path, so that the driver can reach his/her destination with minimum energy consumption. Also, these models have considered speed, acceleration, jerk and road related parameters only but there are many other influencing parameters that also need to be considered.

In brief, a number of different approaches (rule-based and data driven) for energy consumption estimation are discussed. Table 2.1, summarizes these techniques based on a number of different parameters. In the table, four different input factors namely, dynamics, traffic, environment and auxiliary, are considered for comparing the techniques. The input factors related to dynamics include vehicle speed, acceleration and road grade etc. Similarly, factors related to traffic (such as traffic congestion, historical traffic data etc.), environment (such as wind speed, wind direction and temperature etc.) and auxiliary (such as AC/heater, power brakes and lights etc.) are considered. From the table, it can be concluded that most of these models use only a subset of these influencing factors for estimating the energy consumption of EV.

2.2 Techniques for Traffic Prediction

Traffic is an important factor to consider when estimating the energy consumption of an EV with a specific level of accuracy. Traffic affects the driver behaviour while driving. Normally, people drive the vehicle with a desired speed depending on traffic levels. Congested traffic

Table 2.1: Summary of energy consumption estimation techniques for EVs

Year	Input factors				Data Source	Modeling Scale	Reference
	Dynamics	Traffic	Environment	Auxiliary			
2014	✓	×	×	×	Simulation	Microscopic	[44]
2015	✓	×	×	×	Real-world	Macroscopic	[75]
2015	✓	×	✓	×	Real-world	Macroscopic	[76]
2015	✓	×	✓	✓	Real-world	Microscopic	[63]
2015	✓	✓	✓	✓	Real-world	Microscopic	[65]
2015	✓	×	×	×	Real-world	Microscopic	[58]
2015	✓	×	×	×	Real-world	Microscopic	[59]
2016	✓	×	×	✓	Dynamometer	Microscopic	[60]
2016	✓	×	✓	✓	Real-world	Microscopic	[64]
2017	✓	×	✓	×	Simulation	Microscopic	[53]
2017	✓	×	×	×	Simulation	Microscopic	[54]
2017	✓	×	×	×	Dynamometer	Microscopic	[66]
2017	✓	×	✓	✓	Real-world	Macroscopic	[67]
2017	✓	×	×	✓	Simulation	Microscopic	[68]
2017	✓	×	×	✓	Real-world	Macroscopic	[69]
2018	✓	×	×	×	Real-world	Microscopic	[61]
2018	✓	×	✓	✓	Real-world	Macroscopic	[72]
2018	✓	✓	×	×	Real-world	Mesoscopic	[70]
2018	✓	✓	×	×	Real-world	Macroscopic	[73]
2019	✓	✓	×	✓	Simulation	Microscopic	[62]
2019	✓	×	×	×	Real-world	Microscopic	[74]
2020	✓	✓	✓	×	Real-world	Macroscopic	[77]
2020	✓	✓	×	×	Real-world	Macroscopic	[71]
2020	✓	×	×	✓	Simulation	Microscopic	[57]
2020	✓	✓	✓	✓	Simulation	Microscopic	[55]

makes the driver apply brakes again and again whereas when traffic is light the driver can drive the vehicle more freely. It is very difficult to predict the future traffic speed for a particular route as it depends on a number of factors such as current traffic levels, traffic conditions and time of the day etc. There are number of different techniques proposed by researchers in the literature to predict traffic. These techniques can be divided mainly into two categories namely, parametric and non-parametric techniques, which can further be classified into different categories, as shown in Figure 2.4. The following sub-sections discuss the various techniques found in the literature for traffic prediction based on these categories.

2.2.1 Parametric Traffic Prediction Techniques

Parametric Techniques use a fixed number of parameters for representing the behaviour of the data. There are number of approaches developed by different researchers using parametric techniques for predicting traffic in future. Most of these techniques use time series analysis such

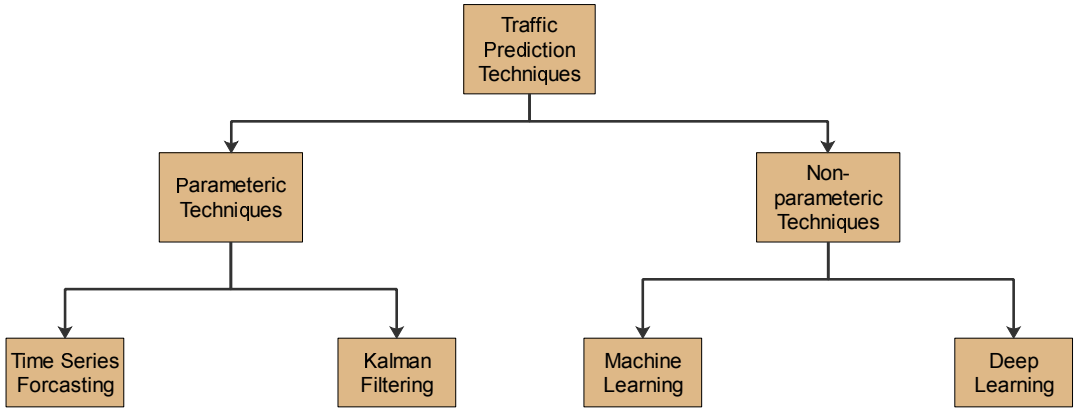


Figure 2.4: Classification of traffic prediction techniques

as Auto-Regressive Integrated Moving Average (ARIMA) [79, 80], ARIMA with explanatory variable (ARIMAX) [81], Seasonal ARIMA [82, 83], Vector ARIMA [84] etc. Other than this, statistical models such as Kalman filter has also been used in number of studies for traffic prediction [85–87].

2.2.1.1 Traffic Prediction using Time-series Forecasting Techniques

Among traffic prediction approaches based on time series analysis the most commonly used approach is ARIMA and its different variants such as ARIMAX, Seasonal ARIMA (SARIMA) and Vector ARIMA (VARIMA) etc. The ARIMA model is an extension of Auto-Regressive Moving Average (ARMA). ARIMA is mainly used to predict future values based on the past values of a time-series and is represented as $ARIMA(p, d, q)$. Here, p, d and q represent autoregressive, integrated and moving average polynomial order, respectively. The following paragraphs discuss the different time series approaches developed for predicting traffic.

Levin and Tsao [79] proposed an Box-Jenkins's ARIMA based approach to predict short-term traffic volume. For this, the traffic volume and occupancy data was collected from two freeway locations in Illinois, USA. They found that the $ARIMA(0, 1, 1)$ model provided statistically better predictions for both traffic volume and occupancy. The developed model gives short-term traffic prediction with forecasting durations of 20, 40 and 60 seconds. Similar to this, Hamed et al. [80] implemented the Box-Jenkins's ARIMA model for predicting short-term traffic volume for next 60 seconds. They also have used the $ARIMA(0, 1, 1)$ to predict traffic volume based on past values. Both the models developed in [79] and [80] are similar. However, the ARIMA model developed in [79] was tested on data from expressways whereas the model in [80] was tested on data from urban arterial roads. Also, both traffic volume and occupancy can be predicted using the former ARIMA model considering historical volume and occupancy data whereas the later model was developed to predict traffic volume only.

Williams [81] used ARIMAX for short-term traffic flow prediction. The proposed ARIMAX model was applied to the traffic data obtained from freeways of France. The multivariate ARIMAX model takes traffic flow data of multiple upstream sensors to forecast the traffic flow at target sensors. The proposed model was compared with other univariate traffic prediction models and it has been observed that the proposed model provide better prediction with less error.

To consider the seasonal variations of traffic data Williams and Hoel [82] developed a traffic prediction model using seasonal ARIMA. The seasonal ARIMA model was applied on traffic data collected from two freeway locations in United Kingdom and United States. In U.K. traffic data was collected from the M25 motorway around London using paired inductive loops and the traffic data of U.S. was collected from video cameras on I-285 freeway. The proposed model was applied to consider seasonal differences with one-week lag and in comparison with other ARIMA based techniques it has been concluded that the proposed model was able to capture the seasonal variation more effectively.

Kumar and Vanajakshi [83] used seasonal ARIMA for short-term traffic flow prediction. They proposed a method to solve the problem of large data requirement of ARIMA based model. For this, traffic flow data was collected using an automated traffic sensor named Collect-R camera for three consecutive days (Sept 20, 2012 to Sept 22, 2012) from Rajiv Gandhi road of Chennai, India. Although, the comparison results show the effectiveness of the proposed model but the proposed model was tested on very small dataset. Also, the effect of linked roads has not been considered in the prediction.

Chandra and Al-Deek [84] also used a variant of ARIMA i.e. Vector ARIMA for short-term traffic speed prediction. Instead of using a univariate model, a multivariate model was proposed considering the effect of upstream and downstream locations on target location. The proposed model was developed using traffic speed data from five stations on I-4 corridor in Florida. The comparison of the results showed that VARIMA performed much better than traditional ARIMA models.

In brief, the above discussed ARIMA based models use time-series analysis to predict future traffic based on the past traffic data. The main problem with these models is that the ARIMA based models work on the assumption of stationary data i.e. the mean and variance of the data does not change. This is not the case with traffic data as this data exhibits non-linear properties and depends on multiple other factors such as connected roads, the environment conditions and time of day etc. Also, these models are sensitive to noise which makes the use of these models in real-life unrealistic because data collected from different sensors is susceptible to noise in sensors due to interference.

2.2.1.2 Traffic Prediction using Kalman Filtering

Another popular technique in parametric techniques is Kalman Filtering which is generally used for non-stationary data. The following paragraphs discuss previous work to predict traffic based on Kalman Filtering.

In 1984, Okutani and Stephanedes [85] used Kalman filtering to predict traffic volume. For this, the traffic volume data was collected from the street network of Nagoya, Japan. For predicting traffic volume on a street link they have considered the input data from the number of connected links. The proposed model performed well for predicting traffic volume with an average prediction error less than 9%. Xie et al. [86] proposed a short-term traffic volume prediction model using Kalman Filter and discrete wavelet transformation. The wavelet transformation was used to remove noise from the data by dividing it into several detailed and approximate data. The Kalman filtering is then applied to predict the future traffic volume using the de-noised data. The proposed model was applied on the traffic data collected from four different locations in Seattle using loop detector sensors. The results of the comparison showed that the proposed model outperforms the approaches with direct Kalman filtering implementation. Ojeda et al. [87] proposed four models using adaptive Kalman filter for predicting traffic flow multiple time-steps ahead. For this, traffic data was collected for 24 hours from Grenoble south ring road in France. The proposed approach was compared with other approaches and it performed better than others for all test scenarios.

2.2.2 Non-parametric Traffic Prediction Techniques

Recent advancement in computing power and data management has increased the popularity of non-parametric techniques for traffic prediction. They are different from parametric techniques, as they do not use a fixed number of parameters and also they do not assume the stationarity of data. Based on the type of technique used, these can be further classified into machine learning and deep learning techniques. The following sections discuss the various approaches proposed by different researchers using non-parametric techniques.

2.2.2.1 Machine Learning based Techniques for Traffic Prediction

Machine learning algorithms such as artificial neural network, support vector machine etc. are being used by different researchers [88, 89] to solve a number of different problems such as object recognition, gesture recognition etc. Due to their increased popularity and ability to learn non-linear patterns, several machine learning algorithms also have been proposed for traffic forecasting and these algorithms have shown promising improvement over statistical

techniques. Among many machine learning algorithms, the three algorithms namely, k-Nearest Neighbor (kNN), Support Vector Regression (SVR) and Artificial Neural Network (ANN) are mainly used for the traffic prediction problem.

Davis and Nihan [90] proposed the use of kNN for short-term traffic forecasting to overcome the limitations of parametric approaches. They compared the proposed kNN approach for freeway traffic forecasting with other univariate linear time-series approaches and found that kNN performed comparably to the linear time-series approaches. Chang et al. [91] developed a dynamic system using k-nearest neighbor non-parametric regression (kNN-NPR) for short-term traffic flow prediction based on historical data. They tested the model on traffic data collected from Suwon tollgate located on expressway $\#1$ in Korea. The experiments show that the proposed model gave satisfactory prediction even if data values in the time-series vary abruptly.

Wu et al. [92] developed a travel time prediction system using SVR. For this, traffic speed data collected from Sun Yet-Sen Highway of Taipei, Taiwan was used. On comparison with other techniques, it has been observed that due to the capability of SVR to converge rapidly and avoid local minima it performed better for all experiments. Similarly, Su et al. [93] used an online version of SVR namely, incremental SVR for short-term traffic flow prediction. They concluded that the proposed model performed even better than back propagation neural network, in terms of accuracy. Castro-Neto et al. [94] also used online SVR for short-term traffic flow forecasting in typical and atypical conditions. The proposed approach was compared with Gaussian Maximum Likelihood (GML), ANN and Holt exponential smoothing. A comparative analysis concluded that the GML performed better in typical conditions but the proposed online SVR performed better than other models in atypical traffic conditions.

Khotanzad and Sadek [95] used Artificial Neural Network (ANN) and Fuzzy Neural Network (FNN) for network traffic prediction. The output from both neural networks were combined for the final prediction. The proposed model was tested on four different databases containing traffic videos and it has been observed that the ensembled system performed better than the individual ones. Similarly, Csikós et al. [96] also used ANN for traffic speed prediction of urban networks. They have used traffic speed data from a simulated tool called VISSIM by replicating the road intersection of Oktogon square of District 6, Budapest, Hungary. The ANN was used to classify the traffic speed into four different categories based on the traffic data from the simulation.

Other than the above discussed approaches, some researchers also have used Bayesian network [97, 98], decision tree [99, 100] and random forest [101, 102] for traffic prediction. Although, the machine learning models have shown performance improvement over statistical techniques,

these techniques rely highly on manually selected features and these features vary from problem to problem. Also, there are no standard guidelines available for selecting features. Other than this, the above mentioned machine learning models have shallow architectures which limit their capability to extract and understand the highly complex and dynamic patterns from large historical traffic data. So, without proper features set and due to their shallow architectures the machine learning techniques may not be best suited for this complicated traffic prediction task.

2.2.2.2 Deep Learning based Techniques for Traffic Prediction

With the advancement of technology, capability of processing large amounts of data has increased dramatically. Hence, deep learning techniques have gained popularity in the field of computer science and also have been successfully applied in the transportation sector. There are a number of studies carried out by different researchers to predict future traffic using various deep learning techniques. The following paragraph discusses some of these techniques in detail.

Zhang et al. [103] used stacked auto-encoders for predicting traffic congestion in the transport network. For this, they have used time series of snapshots of network traffic congestion maps as input to the model. The stacked auto-encoder model predicted traffic congestion very efficiently but did not consider the effect of other parameters such as traffic flow, occupancy, speed and volume etc. Ma et al. [104] proposed the use of Convolutional Neural Network (CNN) for traffic speed prediction. For this, traffic speed data from various sensors was converted into images before giving them as input to the CNN. Here, they have considered the traffic speed data from nearby sensors located on the same road only and hence did not consider the effect of traffic from neighboring roads. A Deep Belief Network (DBN) based approach has been developed in [105] for traffic flow prediction. The DBN was optimized using the algorithm of multi-objective particle swarm optimization which increase the time for forecasting. Also, the model was trained and tested on a dataset obtained from only nine detectors installed on small section of a highway of Wisconsin so the scalability of the approach has not been tested. Other than these, to consider the temporal dependency of traffic data approaches using Long Short Term Memory (LSTM) has also been developed. For instance, Cui et al. [106] and Liu et al. [107] developed LSTM based approach for traffic prediction. Although, LSTMs can represent the temporal dependencies effectively, they lack the ability to capture spatial dependencies which also plays a vital role in influencing traffic. Hence, for accurate future traffic prediction the model should be capable of capturing both the spatial and temporal dependencies.

In brief, researchers have developed a number of deep learning based techniques for predicting the future traffic. There are some problems associated with the existing studies for instance,

it has been found that with increase in prediction time horizon i.e. when predicting for a time instant in the distant future such as 60-min, the prediction accuracy decreases. Also, the prediction accuracy decreases for traffic prediction in peak hours for most of the deep learning based algorithms. The main reason for this is that most of these studies have either considered spatial dependency or temporal dependency. Also, most of the studies while considering spatial dependency, only considered the traffic information from upstream or downstream sensors and have completely neglected the effect of traffic from other neighboring roads. Although, there are some short-comings, the deep learning algorithms have shown promising results compared to shallow machine learning and statistical algorithms.

2.3 Concluding Remarks

In this chapter, different techniques proposed in the literature have been discussed for estimating energy consumption of EVs and predicting future traffic. The following paragraphs discuss the main findings based on the literature presented in previous sections.

The energy consumption estimation techniques have been divided into two categories namely, rule based and data driven techniques. The rule based techniques developed for energy consumption estimation of an EV require vehicle specific calibration, which in turn require internal vehicle parameters, such as efficiency curve of the motor and internal resistance of the battery etc. These vehicle specific parameters are very difficult to obtain because vehicle manufacturers do not share this information in the public domain. Similarly, the data driven techniques, mostly regression based techniques, require real-world data. As most of these techniques are sensitive to noise, they lack the applicability in the real-world because data is mostly collected using different sensors which provide noisy data due to interference from different sources. As discussed previously, energy consumption of an EV is significantly influenced by a number of factors such as temperature, wind, battery's State of Charge (SOC) and auxiliary loads etc. Most of the data-driven and rule based techniques proposed in the literature use only a subset of these factors. So, to obtain statistical significant estimates it is important to consider the effect of all these factors.

Similarly, the researchers have proposed number of techniques for traffic prediction using time-series forecasting, machine learning and deep learning. The simple time-series models for traffic prediction work mainly on the assumption of stationary data, which is not consistent with the traffic data. Also, as the future predictions greatly depend on the previously observed values the error can propagate to multiple steps in multistep prediction. So, these simple time series

models lack the capability to satisfy the high level of accuracy required for successful application in the real-world. Similarly, the machine learning algorithms proposed in the literature rely heavily on manual feature selection as they can not extract their own features from the raw data. Also, shallow architectures can not extract and understand the complex and ever changing patterns from past traffic data. Also, most of the approaches used for traffic prediction consider only one type of dependencies i.e. either temporal or spatial. Due to this, it has been observed that most of these techniques fail to provide predictions with acceptable level of accuracy when multi-step ahead prediction is required.

Considering the above findings, two deep learning based models for estimating energy consumption have been developed in Chapter 3 and 4. These models are developed to overcome a number of research gaps: (i) the models do not require internal vehicle parameters for making predictions, (ii) they are robust towards noisy data, and (iii) they consider the effect all the influencing factors such as road elevation, vehicle speed, acceleration, wind speed, environmental temperature, auxiliary loads and the battery's SOC. In Chapter 5, a multi-step traffic speed prediction model has been developed. This model addresses two main research gaps: (i) unlike other machine learning techniques the developed model is capable of selecting its own features, and (ii) provides better traffic predictions considering both spatial and temporal dependencies.

Chapter 3

Basic Energy Estimation (BEE) Model [1]

Energy consumption of EVs depends on various factors such as vehicle characteristics, vehicle speed, road elevation, auxiliary loads and acceleration etc. In real life, these factors vary a lot and hence make the estimation of energy consumption a complex problem. To overcome this problem, in this chapter, a Deep Convolutional Neural Network (D-CNN) based methodology namely, Basic Energy Estimation (BEE) model, has been developed. To the best of author's knowledge, the proposed D-CNN based approach for power/energy estimation of EV is being developed here for the first time. One of the main challenges with existing techniques was the requirement of internal vehicle data from the manufacturers for calibration of simulation models, which is vary hard to obtain as the manufacturers do not share the data in public domain. The BEE model requires only three parameters namely, vehicle speed, road elevation and tractive effort. Also, the required input parameters can be easily obtained or calculated, for instance, vehicle speed and road elevation can be easily obtained using Global Positioning System (GPS) and Geographic Information System (GIS), respectively. Similarly, tractive effort can be calculated easily using equation (3.1), as discussed in Section 3.1, which requires very basic parameters such as linear acceleration (calculated from speed) and vehicle weight (readily available) etc. Along with vehicle weight, the weight of passengers and luggage can also be taken into consideration. In theory the extra weight will influence the energy consumption of an EV significantly, as with more weight more energy is required to propel the vehicle. The analysis of the rate of variance of the energy consumption of an EV, to the extra weight, though interesting is not in the current scope of this thesis. There are some ANN based approaches [75, 76] which provide very promising results and do not require internal vehicle data from the manufacturer but they do not provide real-time output and hence are not useful in the real-world as they can not be used to guide the driver in real-time. In contrast to this, the proposed deep learning based solution provides an energy consumption estimate in real-time as output and hence, remaining driving range of the vehicle. Also, the deep learning architectures can learn more complex patterns than shallow networks, as existing ANN based models have only one hidden layer. Recent advances in computing power and fast learning algorithms have

[1] The content of this chapter is published as "Estimation of energy consumption of electric vehicles using Deep Convolutional Neural Network to reduce driver's range anxiety," *ISA Transactions*, vol. 98, pp. 454–470, 2020. DOI: https://doi.org/10.1016/j.isatra.2019.08.055 (SCI Impact Factor: 4.305)

made training deep learning architectures feasible. For this reason, deep learning architectures have gained a lot of interest in the automotive sector and have been successfully applied to numerous problems such as image classification, object detection, traffic flow prediction etc [108–115]. Also, the nonlinearity and complexity induced by the combination of all the influencing parameters makes the problem of energy consumption estimation more suitable for a deep learning approach, in contrast to other regression techniques. This motivates the authors to focus on the deep learning based models to solve the problem of estimating energy consumption of EVs.

This chapter has been divided into five sections. Section 3.1, describes the datasets used for training and testing the BEE model. The architecture and working of the BEE model is discussed in Section 3.2. Further, in Section 3.3 the results of training and testing are discussed. The BEE model is compared with other existing techniques in Section 3.4. Finally, Section 3.5 concludes the chapter by providing a brief summary. Note that, unless otherwise stated auxiliary load is in W, temperature in °C, time in seconds, speed in m/h, vehicle weight in kg, power in KW and energy in MJ.

3.1 Datasets Description

Data from two different sources for an EV namely, Nissan Leaf 2013, was used for training, validating and testing the BEE model.

One dataset was obtained from Downloadable Dynamometer Database (D^3) [56] generated at the Argonne National Laboratory (ANL) of Advanced Powertrain Research Facility (APRF), under the funding and guidance of the U.S. Department of Energy. It contains data from several dynamometer tests conducted on various EVs at road grade of 0% for several drive cycles.

As this dataset was quite small and not enough for training, validating and testing the BEE model, another dataset was generated from a simulation model of Nissan Leaf 2013 available in FASTSim (Future Automotive Systems Technology Simulator) [50]. The simulation model is based on internal vehicle parameters of Nissan Leaf 2013 [116, 117], given in Table 3.1. Similar to this, simulated models for other EVs can also be developed based on the availability of manufacturer data such as motor efficiency curve, battery internal resistance etc. Using the simulated model of Nissan Leaf 2013 the data was generated for 80 standard drive cycles (such as US06 Supplemental Federal Test Procedures (SFTP), Urban Dynamometer Driving Schedule (UDDS), Highway Fuel Economy Test (HWFET) and New European Driving Cycle

(NEDC)), which also have been widely used by other researchers [53, 68]. In addition, 30 road grade profiles by varying the road grade from -20% to 20% were generated. It is to be noted that for checking the robustness of the BEE model a number of other custom generated road grade profiles or drive cycles also can be used.

Table 3.1: Parameters of Nissan Leaf used for simulation model in FASTSim [116, 117]

Component	Parameter	Value
Motor	Type	Permanent Magnet AC Synchronous
	Max. Power (kW)	80
	Max. Torque (Nm)	253
Transmission	Type	Single Speed
	Final Drive Ratio	7.9
Battery	Type	Lithium Ion
	Number of Cells	192
	Cell Configuration	2 Parallel, 96 Series
	Nominal Cell Voltage (V)	3.7
	Nominal System Voltage (V)	364.8
	Rated Pack Capacity (Ah)	66.2
	Rated Pack Energy (kWh)	24
Vehicle	Front & Rear Track (m)	1.53
	Vehicle Weight (kg)	1498
	Drive Train	Front Wheel Drive
	Aerodynamic Drag Coefficient	0.29
	Frontal Area (m^2)	2.27
	Wheelbase (m)	2.7
	Weight Distribution Front/Rear (%)	58/42
	Wheel Radius (m)	0.3162

Henceforth, in this chapter the dataset generated through the simulation model and dataset obtained from the Downloadable Dynamometer Database will be referred as $DS-I$ and $DS-II$, respectively. As both the datasets are obtained for the same EV named Nissan Leaf 2013 and also a number of drive cycles are common in the datasets, so they are comparable. Training and validation of the BEE model was carried out using dataset $DS-I$, while $DS-II$ was used for testing the BEE model. This is done to check the developed model's robustness and its ability to generalize out of distribution samples. This does not create any biasing as test data is unseen by the trained model. The datasets $DS-I$ and $DS-II$ both contain data recorded at 10 Hz frequency i.e., 10 readings for every sec. The dataset $DS-I$ contain various parameters such as vehicle speed, battery power supplied, battery's state of charge, environmental temperature, tractive effort, road elevation and auxiliary loads. In the current work, for training, validating and testing the BEE model four parameters were selected, namely Vehicle Speed (v_{sp}), Tractive Effort (t_{eff}), Elevation of the road (r_{el}) and Power Supplied by battery (p_{batt}) at environmental temperature of 25°C and constant auxiliary load of 150 W.

The effect of auxiliary load on energy consumption is additive in nature, so does not increase the complexity of the problem. Also, the environmental temperature does not affect the energy consumption of EV significantly, unless there is a huge change in climatic temperature. So, this dataset, although recorded at $25°$C, is valid for wide range of temperature. The three parameters v_{sp}, r_{el} and p_{batt} are straight forward but t_{eff} refers to the driving force required by the vehicle to move forward which is a combination of multiple components and can be calculated using the following equation, provided in [53]:

$$t_{eff} = f_{ad} + f_{rr} + f_{hc} + f_{la} + f_{wa} \tag{3.1}$$

where f_{ad} represent the opposing force due to aerodynamic drag, f_{rr} is the opposing rolling resistance force, f_{hc} is the gravitational force component which acts while climbing a hill, f_{la} is the opposing force due to linear acceleration and f_{wa} is the inertial force due to rotating parts of the vehicle. So, tractive effort t_{eff} contains the combined effect of all these forces, which in turn depend upon a number of vehicle characteristics, such as frontal area of the vehicle, vehicle's aerodynamic drag coefficient, vehicle's mass and rolling resistance coefficient etc. Due to this, tractive effort t_{eff} along with road elevation r_{el} and vehicle's speed v_{sp} are the ideal candidate to be considered for input and instantaneous power supplied by battery p_{batt} for output.

3.2 Architecture of BEE Model

Energy consumption of an EV depends upon number of factors such as road elevation, auxiliary loads, vehicle speed and vehicle acceleration etc. These factors have a non-linear relation among them as they vary a lot in the real-world. So, to provide an estimate with acceptable statistical confidence this non-linearity is dealt with by adopting the deep learning based methodology.

Deep learning architectures are capable of learning high dimensional non-linear functions using a sequence of semi-affine non-linear transformations. The deep architectures can be represented as a graph of nodes and edges. Each edge has a weight which signifies the relative importance of the link and each node applies an activation function to the weighted sum of incoming connections. A number of activation functions are available such as sigmoid function, tanh etc. A particular deep learning architecture, namely, Convolutional Neural Network (CNN), has been used in this work for the estimation of energy/power consumption of EV.

The CNN has a unique learning ability from images due to its two unique characteristics, namely, pooling mechanism and locally connected layers. The pooling mechanism significantly reduces the number of parameters required for training the network while preserving the important features. In locally connected layers, the output neurons of the layers are connected to their local input neurons only instead of all the input neurons, as in fully connected layers. This helps CNN to effectively extract the critical local features from the images because every layer tries to extract a different feature of the prediction problem.

Considering the above-mentioned characteristics, an image based CNN was chosen and used for estimating the energy consumption of EV. Figure 3.1 represents the complete architecture of the BEE model. There are mainly two modules namely, Time Series to Image Encoder and Image based Deep Convolutional Neural Network.

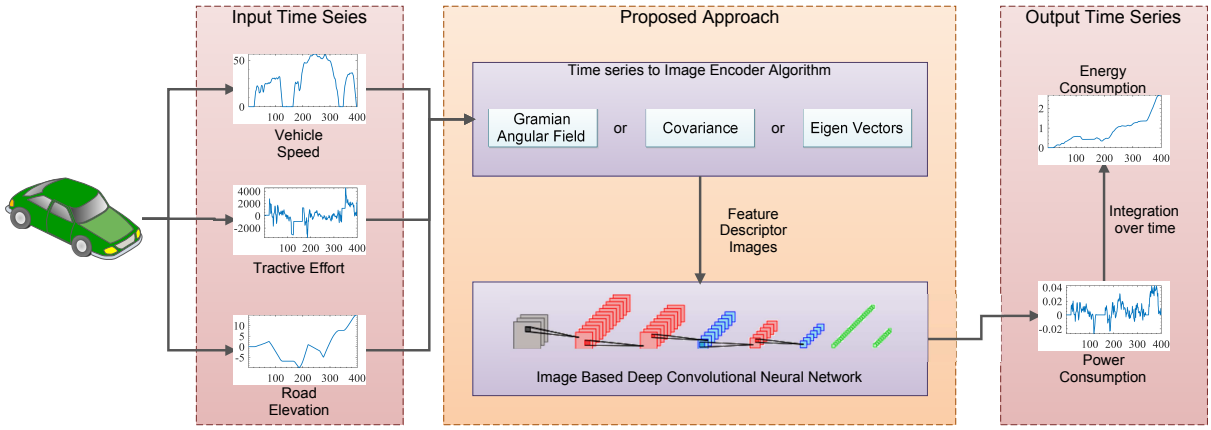


Figure 3.1: Architecture of the Basic Energy Estimation (BEE) model

3.2.1 Time Series to Image Encoder

While there are recurrent neural networks for time-series classification, some researchers have also considered the transformation of time-series into a 2D signal thus taking advantage of a CNN based classification or regression. CNN models have proved their performance for recognizing patterns from images. Hence, to take advantage of the success of CNN models in learning features from images, literature was explored for existing algorithms to convert time series data into images. Also, different features in an image representation of a time-series data, not present in its 1D form, elevates the performance of the task. A number of approaches have been proposed by researchers for encoding time series data to images, for instance, Yang et al. [118] have proposed a method, to encode time series data to images for human activity recognition, in which the multiple time series were concatenated as rows of image i.e. each time series correspond to the particular row in the image. This method is not suitable for the current

problem, as only three parameters, namely, road elevation, the speed of the vehicle and tractive effort have been considered for the input. So, images with only three rows are not appropriate for training the CNN models. In 2015, Wang and Oates [119] proposed Gramian Angular Field (GAF) and Markov Transition Field (MTF) as two approaches to encode time series data to images for classification. However, a lot of information is found to be lost using MTF because the time series need to be binned to a number of quantile bins. Hence, for our work, it is very difficult to even roughly recover the original signal after applying MTF whereas in GAF the information loss is comparatively lower, i.e. it is possible to approximately reconstruct the original signal.

Hence, in this work, GAF has been used as one of the approaches to convert time series data to images. Due to some information loss in GAF, the covariance and eigenvector methods also were considered for conversion. The covariance descriptor reflects the correlation information, hence accommodating the power consumption changes due to instant acceleration. Also, the covariance matrix being symmetric becomes computationally effective. The eigenvectors of a covariance matrix give a set of orthonormal vectors which indicate the directions in which the data varies the most (principal components). In general, CNN uses augmentation techniques (such as Principal Component Analysis and whitening) to reduce overfitting. Hence, motivating the use of eigenvectors as feature input.

In order to convert time series data into images, the selected time series namely, Vehicle Speed (v_{sp}), Tractive Effort (t_{eff}), Elevation of the road (r_{el}) and Power Supplied by battery (p_{batt}), from dataset $DS-I$ and $DS-II$ were partitioned into m small time series each of 10 sec duration, such that dataset $DS-I$ contain approximately 3.5 lacs while $DS-II$ contains approximately 3500 partitions. Out of these, 70% of the partitioned time series were randomly selected from $DS-I$ (say $DS-I_{tr}$) and were used for training and rest 30% (say $DS-I_{val}$) were used for validating the CNN models. As discussed previously in section 3.1, out of the four time series, the first three were used as input to the CNN model and the fourth one was taken as output. So, the partitions of input time series only were converted into images using three preprocessing techniques namely, GAF, Covariance and Eigenvectors, and generate three different sets of images as output. The output sets can be represented, in general, using equation 3.2.

$$\mathbb{X} = \{M^i \mid M^i \in \mathbb{R}^{100\times100\times3} \; and \; i = 1, 2, ..., m\} \tag{3.2}$$

where $\mathbb{X}$ is the output set obtained after using particular preprocessing technique, M^i's are the images obtained from corresponding i^{th} partition of input time series of v_{sp}, t_{eff} and r_{el}, as

shown in Figure 3.2. In this figure, the input signals were the i^{th} partition of time series v_{sp}, t_{eff} and r_{el} (highlighted in red) and denoted by $v^i_{sp}\_In$, $t^i_{eff}\_In$ and $r^i_{el}\_In$. The preprocessing algorithm can be any of the three methods, namely GAF, Covariance and Eigen Vectors. For each input, the preprocessing algorithms gave the corresponding output of size 100×100 , denoted by $v^i_{sp}\_Out$, $t^i_{eff}\_Out$ and $r^i_{el}\_Out$. The output matrices generated were then concatenated to obtain the corresponding image M^i which was then fed to CNN models as input. The three preprocessing methods are discussed as follows:

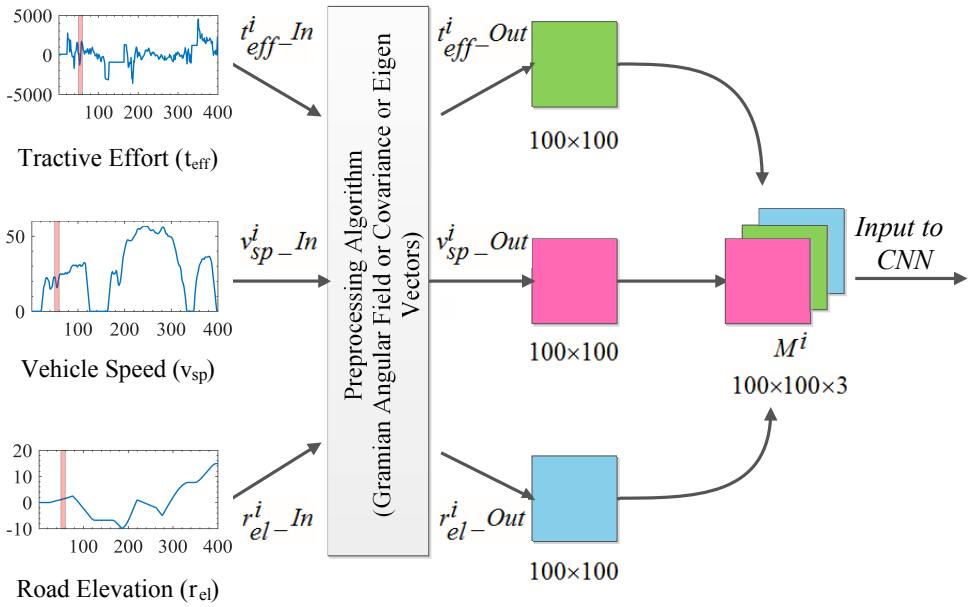


Figure 3.2: Preprocessing of Time Series Data

i) *Gramian Angular Field (GAF):* In Gramian Angular Field method, proposed in [119], initially all the m partitioned time series (say x^i's) of Vehicle Speed (v_{sp}), Tractive Effort (t_{eff}) and Road Elevation (r_{el}) were normalized into the range of [-1,1], using Equation (3.3), to generate three new sets $\hat{V}$, $\hat{T}$ and $\hat{R}$ each containing the corresponding normalized partitions $\hat{x^i} \in \mathbb{R}^{100\times1}$ of v_{sp}, t_{eff} and r_{el} respectively.

$$\hat{x^i} = \frac{(x^i - \max(x)) + (x^i - \min(x))}{\max(x) - \min(x)} \tag{3.3}$$

In this equation, $\max(x)$ and $\min(x)$ represents the maximum and minimum values of time series x. Then, the sets $\hat{V}$, $\hat{T}$ and $\hat{R}$ were further transformed to three new sets V, T and R of GAF matrices $G^i \in \mathbb{R}^{100\times100}$ obtained from corresponding normalized partitions

$\hat{x^i}$, by using Equation (3.4).

$$G^i = \hat{x^i} \cdot \hat{x^i}^T - \sqrt{I - \hat{x^i}^2} \cdot \sqrt{I - \hat{x^i}^2}^T \tag{3.4}$$

where I represents the 1D array $[1, 1, ..., 1]^T$ of length 100. The sets V, T and R were then used to generate the input set $\mathbb{X}$ of images M^is. The j^{th} element of $\mathbb{X}$, i.e. $M^j \in \mathbb{R}^{100\times100\times3}$, was obtained by concatenating j^{th} GAF matrices $G^j \in \mathbb{R}^{100\times100}$ from V, T and R after normalizing them to the range of [0,1], i.e., G^j from V, T and R after normalizing became the first, second and third layer, respectively, of M^j. There is some information loss, as explained in [120], due to the negative term (second term with square roots) in Equation (3.4) which can affect the estimation accuracy of the proposed models.

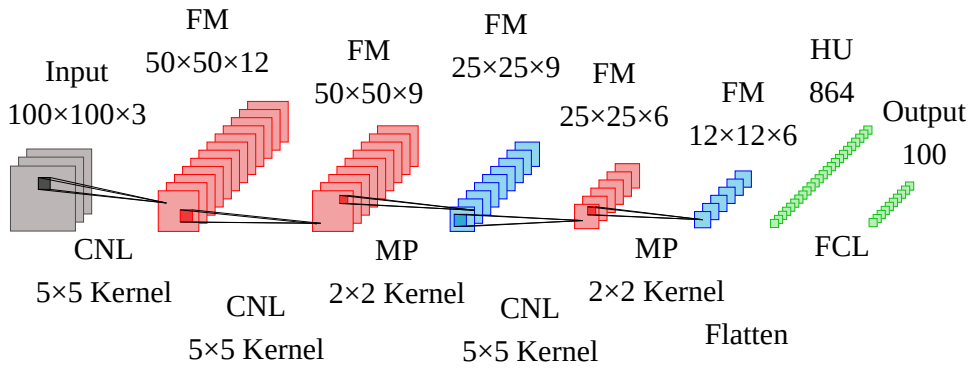
ii) *Covariance:* The loss in information in above method motivated the use of a Covariance matrix as feature input. The first step in this method was to normalize the partitions x^i's of v_{sp}, t_{eff} and r_{el} and obtain three new normalized sets $\hat{V}$, $\hat{T}$ and $\hat{R}$. After normalization, three sets V, T and R were generated each containing the covariance matrices $\hat{C^i} \in \mathbb{R}^{100\times100}$ of $\hat{V}$, $\hat{T}$ and $\hat{R}$ respectively. The sets V, T and R were then used to create the set $\mathbb{X}$ by concatenating the corresponding $\hat{C^i}$'s from V, T and R.

iii) *Eigen Vectors:* In this method, the covariance matrices $C^i \in \mathbb{R}^{100\times100}$ of each partition of v_{sp}, t_{eff} and r_{el} was calculated but without normalization. It generated three sets V^c, T^c and R^c each containing the covariance matrices C^is of v_{sp}, t_{eff} and r_{el}. Then eigen vector matrices E^is from these covariance matrices C^is were calculated and normalized to the range of [0,1]. So three new sets V, T and R were generated each containing the normalized eigen vector matrices $\hat{E^i} \in \mathbb{R}^{100\times100}$. Finally, the set $\mathbb{X}$ was generated by concatenating the corresponding $\hat{E^i}$s from V, T and R.

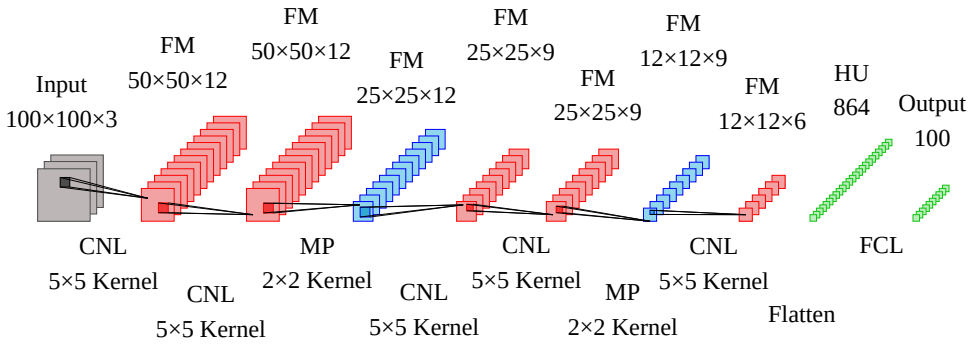
3.2.2 Image based Deep Convolutional Neural Network

CNN architectures have gained a lot of popularity in the field of pattern recognition [121, 122]. AlexNet [123] is one of the most popular and vastly used architecture proposed in the field of pattern recognition. Also, it has been considered as a base reference for researchers applying deep learning in a new domain [124]. Considering the above, initially the authors chose to start with CNN architecture considering AlexNet architecture as the base reference. AlexNet architecture has multiple convolution, pooling and fully connected layers stacked together. So in this work, experiments with multiple CNN architectures, having different number of layers, were performed. Later in the experiments, it has been observed that increasing the layers

further after a particular number of layers (in this case 7) did not enhance the performance for the current data. So, the results for two architectures, as shown in Figure 3.3, are presented in this work. The CNN architecture with seven layers, shown in Figure 3.3a, and architecture with nine layers, shown in Figure 3.3b, are referred to as CNN^7 and CNN^9, respectively.



(a) CNN Model with 7 layers (CNN^7)



(b) CNN Model with 9 layers (CNN^9)

Figure 3.3: Two CNN Models with different architecture used in BEE model (Convolution with Non-Linearity (CNL), Fully Connected Layer (FCL), Max Pooling (MP), Feature Maps (FM), Hidden Units (HU))

CNN^7 architecture takes an image of size $100 \times 100 \times 3$, obtained from Time Series to Image Encoder module, as input and convolves it with 5×5 kernels. The kernels have depth of 3 as the input image has three channels. During the convolution operation, padding of two rows and two columns have been used along with stride of 2 positions. In the first convolution layer, 12 such 5×5 kernels were used which gave an output of $50 \times 50 \times 12$ feature maps (can be calculated using the equation $Output_{size} = ((Input_{size} - Kernel_{size} + 2 \times Padding)/Stride) + 1$). Here, it can be observed that number of channels were increased in multiples of 4 i.e. from 3 to 12 and size of the image has been reduced to $1/4^{th}$ i.e. from 100×100 to 50×50. So, the first convolution layer produces the same number of features as the size of the input image i.e. $100 \times 100 \times 3$ becomes $50 \times 50 \times 12$. The number of kernels for the first layer was chosen as 12 compared to standard sizes of 48 etc in AlexNet for two reasons. First, the data considered is time series data as opposed to more complicated image data. Second, a larger number of kernels

will require a bigger training set in order to converge. The output from the convolution layer was then passed through a non-linear (Tanh) activation layer which maps it using a function $tanh(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$. After the first convolution with non-linearity (CNL) layer, there is a pattern of layers (i.e. one CNL layer then a max pooling layer), which has been repeated twice. This pattern has been used to decrease the dimension and number of feature maps and only keep the important features. For instance, the second CNL layer reduce the number of feature maps from 12 to 9 and then, a max pooling layer has been used which finds the maximum feature map over the local neighborhood and reduce the size of feature maps from 50×50 to 25×25. After the repeated pattern of layers, a flatten layer has been used which changes the shape of feature maps from 3D to 1D because the next layer which is a fully connected layer (FCL) take a 1D vector as input. So, the FCL maps the output of previous flatten layer to the desired output of length 100. Similar to CNN^7 architecture, the CNN^9 architecture was developed by increasing the number of layers. In CNN^9 architecture, the main difference is the number of layers and hence the dimension and number of feature maps decrease slowly. The main reason for this was to keep the important features as long as possible so that more accurate result can be obtained. However, no accuracy gain was observed by increasing the number of layers further after a specific number of layers.

A set of images $\mathbb{X}$, obtained from Time Series to Image Encoder module, was taken as input to the CNN models, and the models generated an output set $\mathbb{Y}$. The output set $\mathbb{Y}$, defined in Equation (3.5), was the set of 1D arrays O^i's each of length 100, corresponding to the instantaneous power supplied by the battery. Each 1D array O^i represents the i^{th} partition of time series p_{batt}, normalized into the range of [0,1].

$$\mathbb{Y} = \{O^i \mid O^i \in \mathbb{R}^{100\times 1} \; and \; i = 1, 2, ..., m\} \tag{3.5}$$

3.3 Experimental Results and Discussion

A number of CNN models with different number of layers were trained with the dataset pre-processed with three methods, namely GAF, Covariance and Eigen Vectors. Henceforth, CNN models with n number of layers trained with GAF, Covariance and Eigen Vector features are denoted as CNN^n_{gaf}, CNN^n_{cov} and CNN^n_{eig}, respectively. In all of these models, 70% of the dataset $DS - I$ (mentioned in Section 3.1) was used for training with 2000 epochs. The remaining 30% of the dataset was used for validation. Furthermore, initially CNN architectures were considered as black boxes and the only performance indicators were the level of accuracy

achieved, error etc but recently Shwartz-Ziv and Tishby [125] have presented an interesting approach to visualize the behaviour of internal hidden layers of Deep Neural Networks (DNN) in an information plane using mutual information of layers. The visualization of internal behaviour of DNN architecture provides the insight into how well the model is training, how many epochs are actually required for fitting (called the drift phase) and whether the particular architecture able to find the fitting solution etc.

3.3.1 Mutual Information of Layers

The mutual information represents the amount of relevant information contained by a random variable X about another random variable Y. The mutual information of any two random variables, X and Y, with joint distribution $p(x, y)$, can be defined as:

$$I(X;Y) = \sum_{x \in X, y \in Y} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{3.6}$$

where $p(x), p(y)$ represent the marginal distribution of the variables X and Y respectively. The mutual information obtained using the above equation range from [0,∞). So for comparison purpose $I(X;Y)$ was normalized to the range of [0,1] by using equation (3.7) as follows:

$$NMI(X;Y) = \frac{I(X;Y)}{\sqrt{H(X)H(Y)}} \tag{3.7}$$

where $H(X)$ and $H(Y)$ represent the entropy of random variables X and Y. A number of other normalizations are also possible based on the observation that $I(X;Y) \leq min(H(X), H(Y))$ using arithmetic or geometric mean of $H(X)$ and $H(Y)$. The geometric mean was used due to the analogy with the normalized inner product in Hilbert Space. As $H(X) = I(X;X)$, it can be observed that $NMI(X;X) = 1$ as desired.

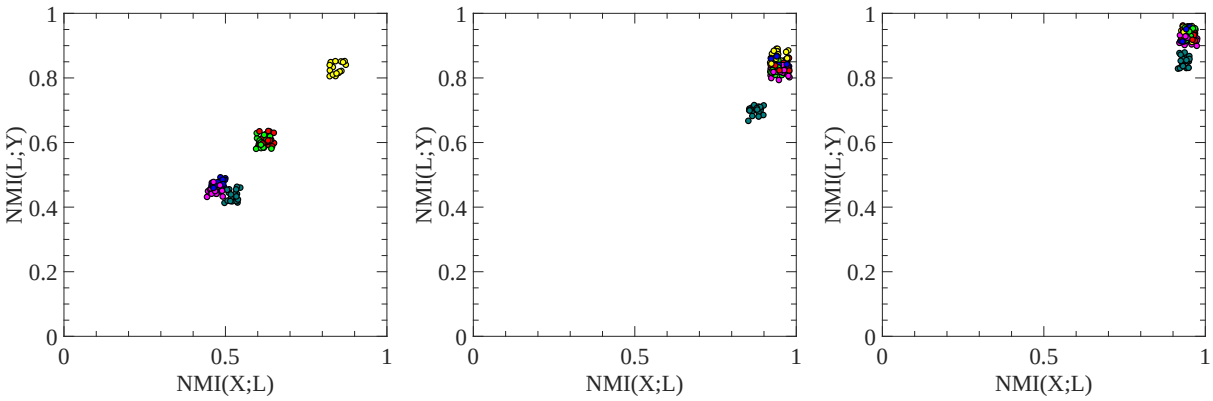
3.3.2 Training and Validation of BEE Models

While training the CNN models, the kernels of each convolutional layer of CNN models were initialized with random numbers generated from a uniform distribution, defined in the range of $[-stdv, stdv)$ where $stdv = 1/\sqrt{kw \times kh \times numInPl}$. Here kw, kh and $numInPl$ represent the kernel width, kernel height and number of input planes of the particular convolutional layer, respectively. The models were trained to learn the kernels for a maximum 2000 epochs using Stochastic Gradient Descent (SGD) with initial learning rate and batch size set to 0.01 and 64, respectively. The learning rate was set to gradually decrease as the training progresses at

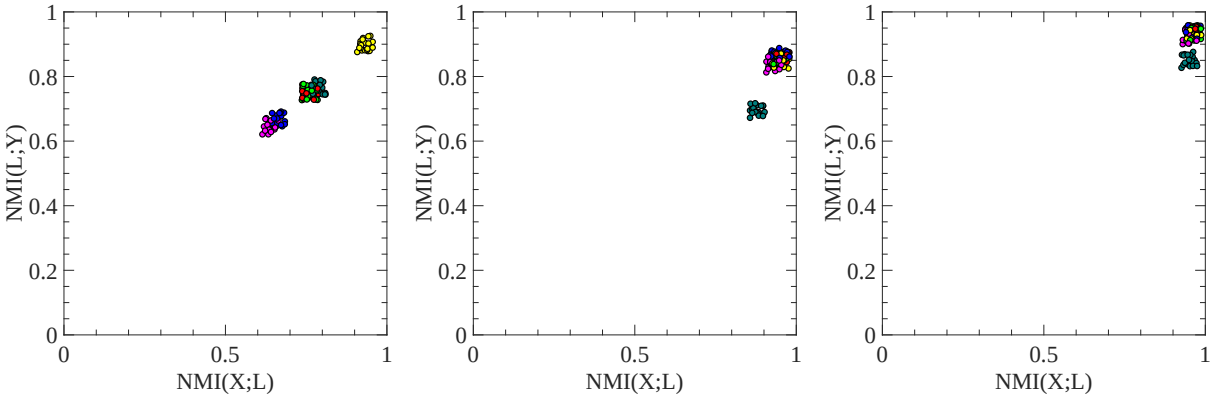
a constant rate. The objective was to minimize the Mean Square Error (MSE) between the predicted and actual power consumption. Experiments were conducted with different number of layers such as 5, 6, 7, 8, 9, 10 and 11. It was found that by increasing the number of layers, the number of epochs to converge reduced, for instance, the CNN models with 5, 7, 9 and 11 layers when trained using dataset preprocessed with the covariance method converged at approximately 420, 380, 330 and 290 epoch, respectively. Although by increasing the layers the models converge early, it is also a well known fact that the architectures with more layers require more training data to achieve the same level of accuracy compared to an architecture with less number of layers. For comparison purpose, the results for two CNN models CNN^7 and CNN^9 are presented in this thesis.

For the experimental purpose, the normalized mutual information for each CNN model was calculated between each layer's output and the model's input i.e. $NMI(X; L^i)$ and between each layer's output and the model's output i.e. $NMI(L^i; Y)$. Here L^i represents the i^{th} layer's output, $X \subseteq \mathbb{X}$ and $Y \subseteq \mathbb{Y}$ represent the input and output of a particular CNN model. For calculating the normalized mutual information, X, Y, and L^i were binned into 5000 equal intervals. Then these discretized X, Y and L^i were used to calculate their joint distributions and hence, normalized mutual information $NMI(X; L^i)$ and $NMI(L^i; Y)$. These calculations were performed repeatedly for 20 randomly initialized CNN models for each CNN architecture trained with 75% of randomly selected training samples. The variations in normalized mutual information for CNN models trained using the dataset $DS - I$ after preprocessing using GAF, Covariance and Eigen Vector methods are shown in Figures 3.4 and 3.5. These figures clearly show that normalized mutual information grows as the training progresses and all the layers starting from different initial state try to obtain the relevant information. The information gain was quite large from initial state until approximately 300^{th} epoch and after that not much further information was gained. So, it is evident from the figures that the networks are training well and can be used even after approximately 300^{th} epoch because after that the layers are just optimizing themselves with minimal information gain. Also, the layers of randomized networks form clusters and behave similarly. So, it is justified to take the average of the randomized networks and plot the average training and validation error across 2000 epochs as shown in Figure 3.6. The training and validation error shown in the figure was calculated from normalized actual and predicted output of the particular CNN model.

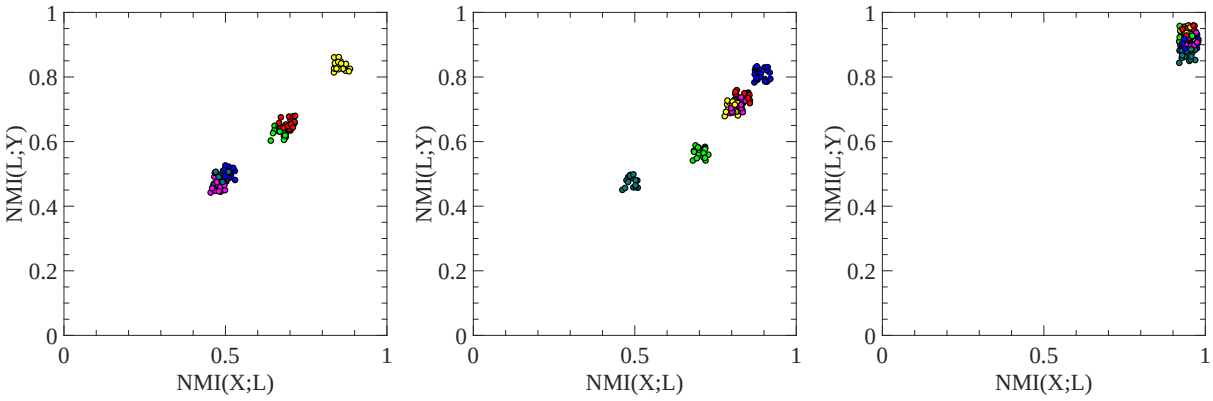
Validation was performed at every 10^{th} epoch. Figure 3.6, shows how the choice of input feature descriptor effects the training and performance of CNN models. It can be clearly observed that the CNN models trained with Eigen Vectors, i.e., CNN^7_{eig} and CNN^9_{eig}, have high training and validation error compared to other CNN models trained with covariance and GAF features. The CNN models trained with covariance feature descriptors outperformed the other models



(a) CNN^7_{cov} at initial state (b) CNN^7_{cov} after 300 epochs (c) CNN^7_{cov} after 2000 epochs



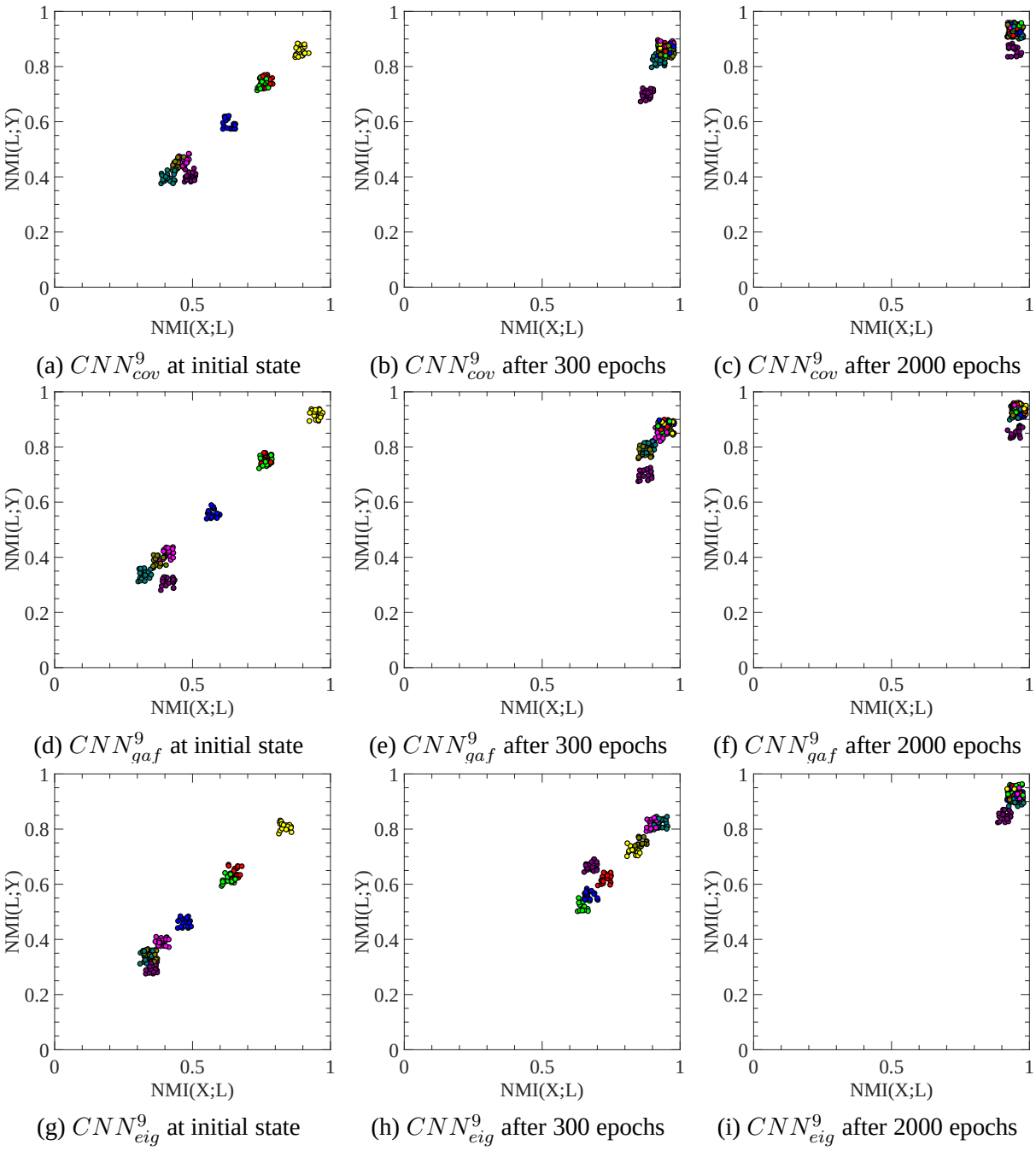
(d) CNN^7_{gaf} at initial state (e) CNN^7_{gaf} after 300 epochs (f) CNN^7_{gaf} after 2000 epochs



(g) CNN^7_{eig} at initial state (h) CNN^7_{eig} after 300 epochs (i) CNN^7_{eig} after 2000 epochs

Figure 3.4: Normalized Mutual Information (NMI) for 20 randomly initialized CNN^7_{cov}, CNN^7_{gaf} and CNN^7_{eig} BEE models. Legend: NMI between input/output and output of (●) Convolutional Layer 1, (●) Convolutional Layer 2, (●) Max Pooling Layer 1, (●) Convolutional Layer 3, (●) Max Pooling Layer 2, (●) Fully Connected Layer

with minimum training and validation error, but out of CNN^7_{cov} and CNN^9_{cov}, which one is better, it is very difficult to conclude from the figure as their training and validation errors are overlapping. Also, it can be observed that the CNN models CNN^7_{cov} and CNN^9_{cov} converged before other models at approximately 300^{th} epoch. All these observations are in accordance



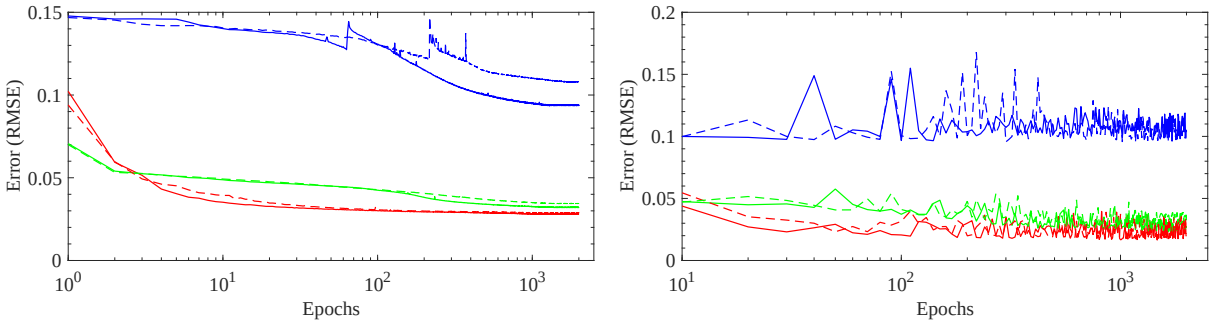
(a) CNN^9_{cov} at initial state (b) CNN^9_{cov} after 300 epochs (c) CNN^9_{cov} after 2000 epochs

(d) CNN^9_{gaf} at initial state (e) CNN^9_{gaf} after 300 epochs (f) CNN^9_{gaf} after 2000 epochs

(g) CNN^9_{eig} at initial state (h) CNN^9_{eig} after 300 epochs (i) CNN^9_{eig} after 2000 epochs

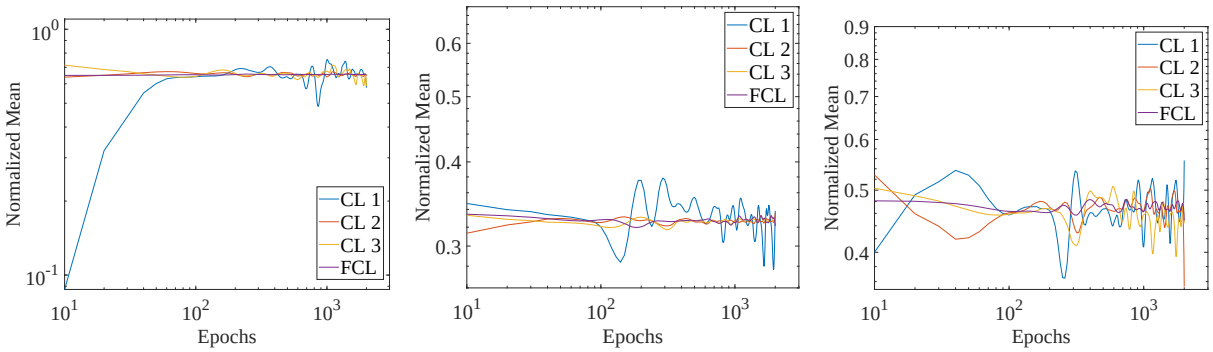
Figure 3.5: Normalized Mutual Information (NMI) for 20 randomly initialized CNN^9_{cov}, CNN^9_{gaf} and CNN^9_{eig} BEE models. Legend: NMI between input/output and output of (●) Convolutional Layer 1, (●) Convolutional Layer 2, (●) Max Pooling Layer 1, (●) Convolutional Layer 3, (●) Convolutional Layer 4, (●) Max Pooling Layer 2, (●) Convolutional Layer 5, (●) Fully Connected Layer

with discussion in section 3.2.1, where it has been explained that the GAF features have some information loss which affect the performance of CNN models trained with GAF. Similarly, the eigen vectors descriptors contain only the direction of variance and loose most of the relevant information compared to the covariance features descriptors.

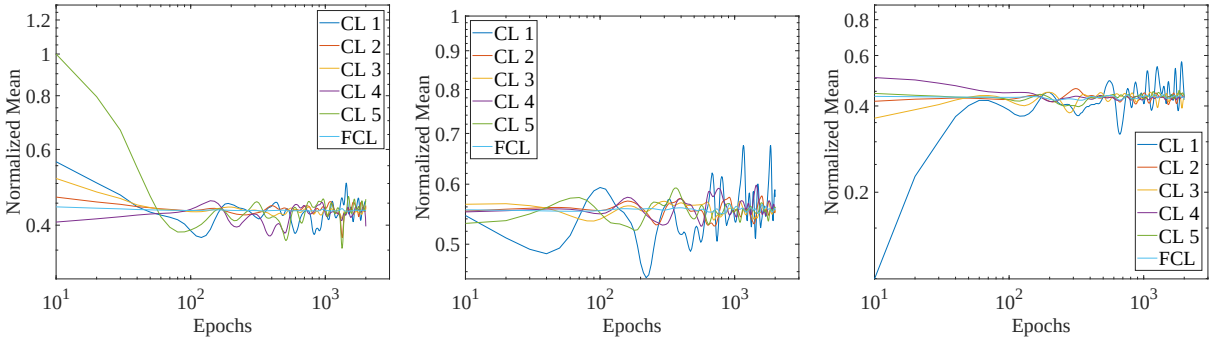


(a) (b)

Figure 3.6: (a) Training and (b) Validation of BEE models across 2000 epochs. Legend: (—) CNN^9_{eig}, (- - -) CNN^7_{eig}, (—) CNN^9_{gaf}, (- - -) CNN^7_{gaf}, (—) CNN^9_{cov}, (- - -) CNN^7_{cov}



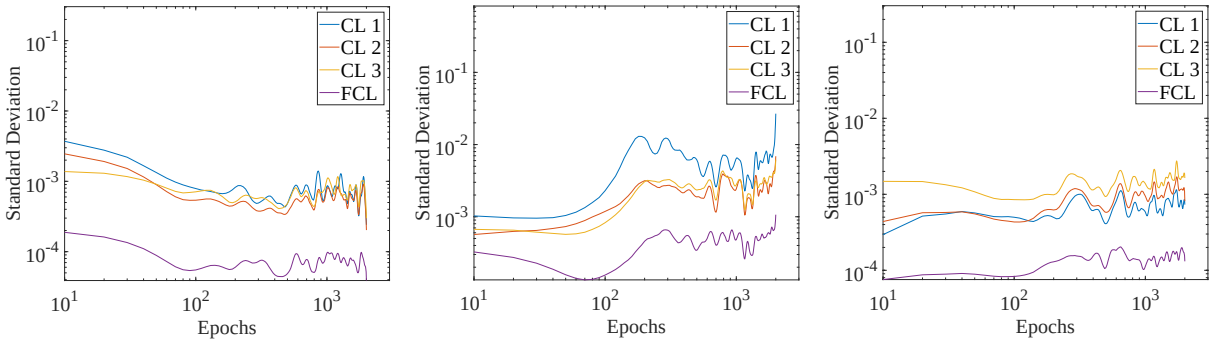
(a) Normalized mean for layers of CNN^7_{cov} (b) Normalized mean for layers of CNN^7_{eig} (c) Normalized mean for layers of CNN^7_{gaf}



(d) Normalized mean for layers of CNN^9_{cov} (e) Normalized mean for layers of CNN^9_{eig} (f) Normalized mean for layers of CNN^9_{gaf}

Figure 3.7: Normalized mean of gradient weights of each layer of BEE models. In legend CL and FCL stands for Convolutional and Fully Connected Layer, respectively

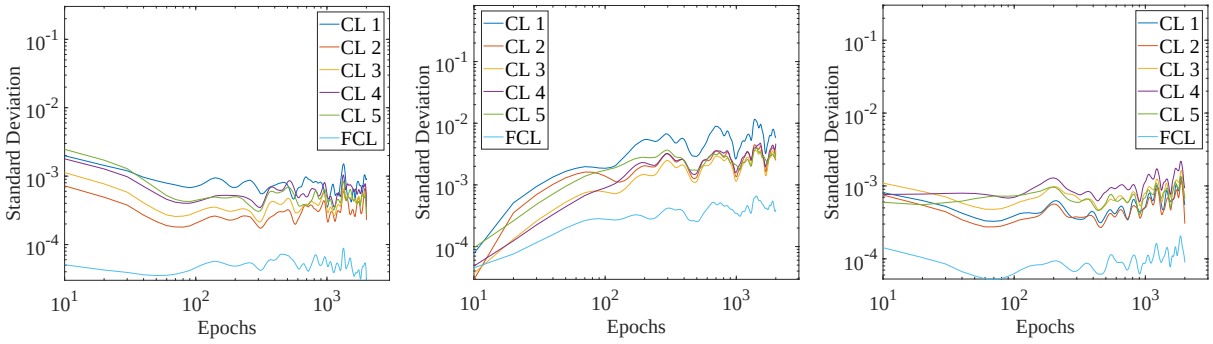
For a better understanding of the behaviour of layers over training, plots of the normalized mean and standard deviations of stochastic gradients of each convolutional and fully connected layer are shown in Figure 3.7 and 3.8. In the figures, the layers are numbered such as CL 1, CL 2 and so on. The lower number represents the layer closer to the input and the number increases on moving towards the output. Also, it can be concluded that the normalized mean



(a) Standard deviation for layers of CNN^{7}_{cov}

(b) Standard deviation for layers of CNN^{7}_{eig}

(c) Standard deviation for layers of CNN^{7}_{gaf}



(d) Standard deviation for layers of CNN^{9}_{cov}

(e) Standard deviation for layers of CNN^{9}_{eig}

(f) Standard deviation for layers of CNN^{9}_{gaf}

Figure 3.8: Standard deviation of gradient weights of each layer of BEE models. In legend CL and FCL represents Convolutional and Fully Connected Layer, respectively

of each layer of each CNN model is converging around a single value. So after converging, the layers are optimizing themselves which can also be observed in Figures 3.4 and 3.5. Another observation is that the mean of the gradient weights are larger than the standard deviation of the gradient weights which indicates small gradient stochasticity which in turn implies high signal to noise ratio (SNR). Also, the difference between the normalized mean and the standard deviation of gradient weights becomes almost constant as training progresses which means that with the training the empirical error saturates. Another observation is that the layers near to the output have lower standard deviation of gradient weights. The main reason for that is the layers near to the output already have a large amount of information regarding the output which results in less deviation in gradient weights. So, it can be concluded that more the information a layer has, the lower is the standard deviation in gradient weight of that layer.

3.3.3 Testing of BEE Models

As discussed previously, the models were tested using the dataset $DS-II$ after preprocessing with the three methods explained in Section 3.2.1. Dataset $DS-II$ has a number of differ-

ent drive cycles to test upon. Figure 3.9 shows the testing results for four such drive cycles namely, UDDS, SFTP, HWFET and NEDC with each CNN model. Figure 3.9 clearly shows that the CNN models trained with eigenvector features i.e. CNN^{7}_{eig} and CNN^{9}_{eig}, in most of the cases, are no-where near the target value which is consistent with the conclusion drawn

from Figure 3.6. The models trained with GAF and covariance feature are really close to the target power consumption. On closer examination, CNN model CNN^{7}_{cov} are found to consistently perform better in all of the cases compared to others. The main reason for this is the amount of information each feature descriptor holds. As explained previously, the covariance feature descriptors hold the maximum amount of information compared to GAF and eigenvectors. To make the above conclusion clear, Table 3.2 shows the percentage energy consumption deviation (calculated using equation (3.8)) for the above four drive cycles by each CNN model compared to actual energy consumption.

$$E_{dev} = \frac{|\int_{t=0}^{T} P_{act}(t)dt - \int_{t=0}^{T} P_{est}(t)dt|}{\int_{t=0}^{T} P_{act}(t)dt} \times 100 \tag{3.8}$$

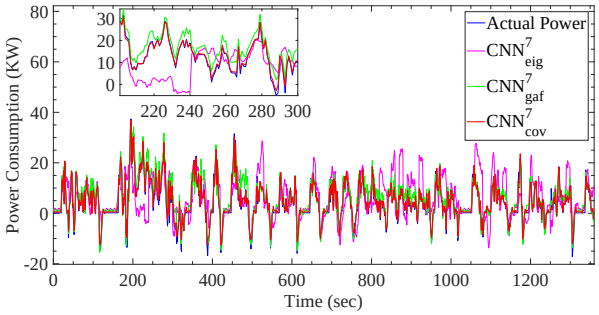
where E_{dev} represents the percentage energy consumption deviation, $P_{act}(t)$ and $P_{est}(t)$ is the actual and estimated instantaneous power consumption at time t, respectively. From Table 3.2, it can be concluded that CNN^{7}_{cov} performed consistently better compared to other CNN models with the lowest energy consumption deviation. There are some exceptions such as in case of HWFET drive cycle CNN^{9}_{cov} performed marginally better than CNN^{7}_{cov}. So, to justify and generalize the above conclusion a cross validation has been performed for all the BEE models and presented in the next subsection.

Table 3.2: Total energy consumption deviation (in percentage) for different drive cycles with BEE models

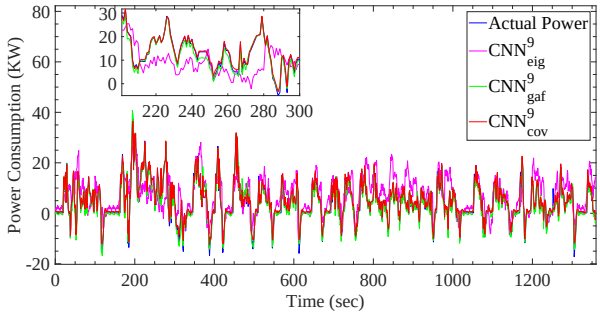
	CNN^{9}_{cov}	CNN^{9}_{eig}	CNN^{9}_{gaf}	CNN^{7}_{cov}	CNN^{7}_{eig}	CNN^{7}_{gaf}
UDDS	13.18	59.56	13.90	7.04	48.09	28.92
SFTP	10.75	48.58	10.68	2.93	36.22	13.47
HWFET	06.31	35.86	07.97	6.61	32.20	12.18
NEDC	11.39	55.90	17.50	6.04	28.94	21.53

3.3.4 Cross Validation of BEE Models

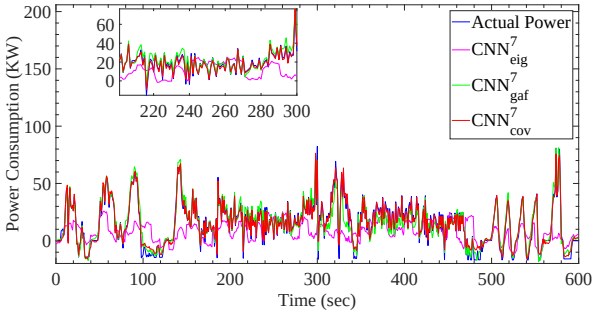
To show the generalization ability and to measure the robustness of the BEE models, cross validation has been performed for all the CNN models. For this, a cross validation technique named k-fold cross validation has been used. The dataset $DS - I$ was partitioned into k



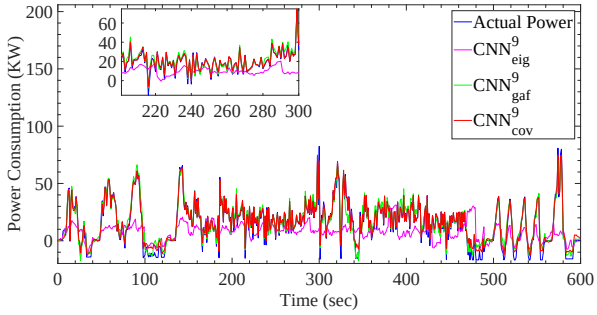
(a) Estimated power consumption by CNN^7 for UDDS drive cycle



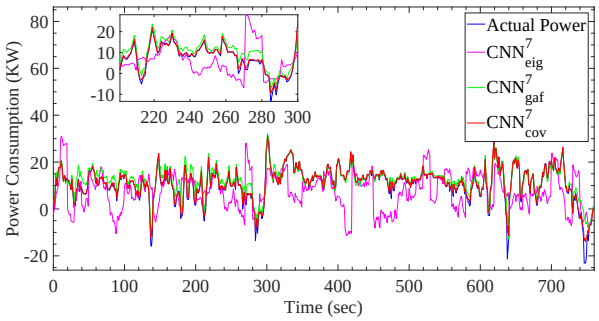
(b) Estimated power consumption by CNN^9 for UDDS drive cycle



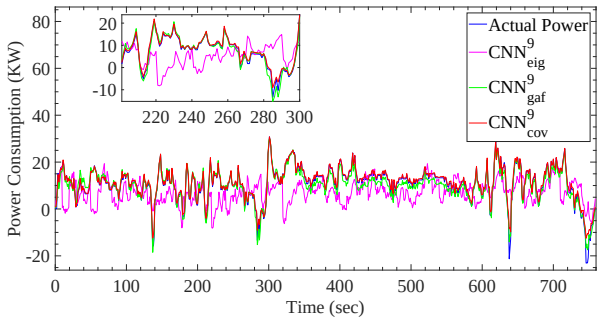
(c) Estimated power consumption by CNN^7 for SFTP drive cycle



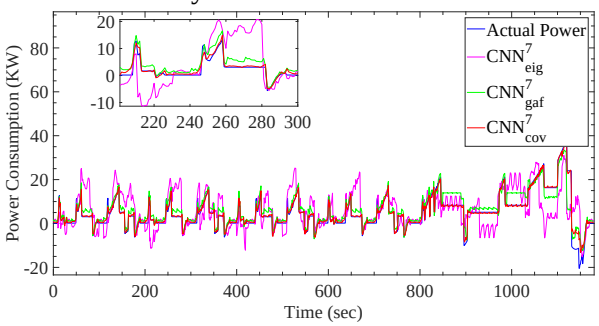
(d) Estimated power consumption by CNN^9 for SFTP drive cycle



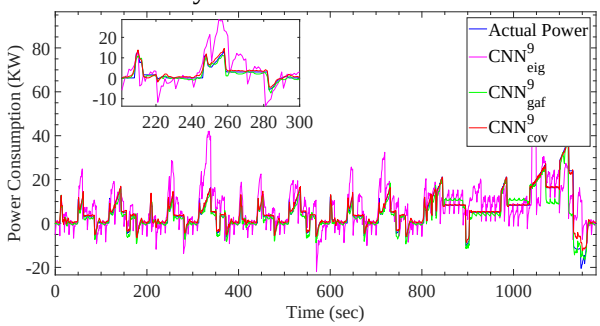
(e) Estimated power consumption by CNN^7 for HWFET drive cycle



(f) Estimated power consumption by CNN^9 for HWFET drive cycle



(g) Estimated power consumption by CNN^7 for NEDC drive cycle



(h) Estimated power consumption by CNN^9 for NEDC drive cycle

Figure 3.9: Estimated power consumption for different driving cycles with BEE models

equally sized partitions. Then, 70% of these k partitions was selected and used for training the

CNN models and remaining 30% was used for validation. This process was repeated k times (the folds), such that each of the k partitions used at least once as part of validation set. The following are the different metrics that have been used as performance indicators.

i) *Root Mean Square Error (RMSE):* RMSE is a very popular and standardized formula to measure the error rate and hence the performance of a system. It can be calculated using the below equation:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(P_{act}^{i} - P_{est}^{i})^2}{n}} \tag{3.9}$$

where P_{act} is the actual power consumption, P_{est} is the estimated power consumption by the CNN model and n is the total number of instances.

ii) *Mean Absolute Error (MAE):* MAE is also a standardized measure which gives the idea of absolute deviation of estimated value with respect to actual value. The following is the equation which was used to calculate mean absolute error between actual and estimated power consumption:

$$MAE = \frac{\sum_{i=1}^{n} |(P_{act}^{i} - P_{est}^{i})|}{n} \tag{3.10}$$

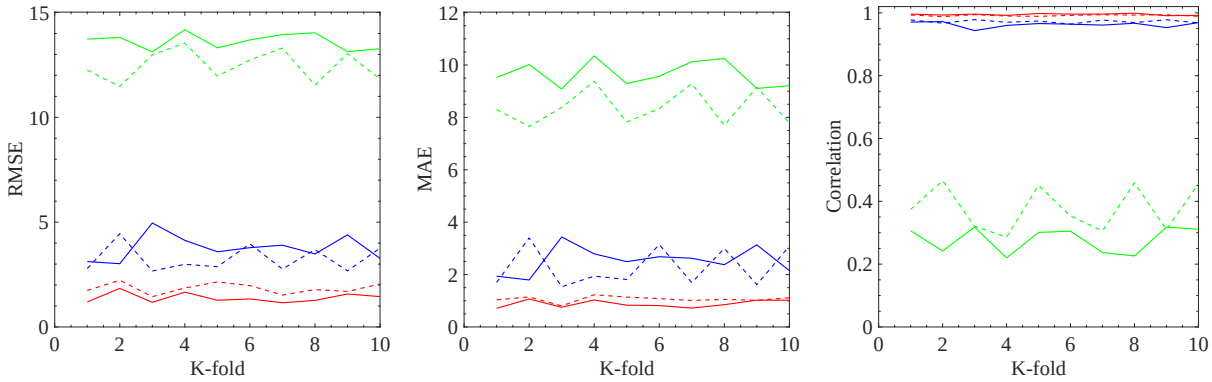
Similar to RMSE, the symbols P_{act}, P_{est} and n represent actual power consumption, estimated power consumption and total number of instances, respectively.

iii) *Correlation (Corr):* Correlation represents the statistical relationship between actual and estimated value. It can be calculated using the below equation:

$$Corr = \frac{\sum_{i=1}^{n}(P_{act}^{i} - \overline{P_{act}})(P_{est}^{i} - \overline{P_{est}})}{\sqrt{\sum_{i=1}^{n}(P_{act}^{i} - \overline{P_{act}})^2 \sum_{i=1}^{n}(P_{est}^{i} - \overline{P_{est}})^2}} \tag{3.11}$$

where $\overline{P_{act}}$ and $\overline{P_{est}}$ represent the mean of the actual power consumption and mean of the estimated power consumption and remaining symbols are the same as RMSE or MAE. The correlation lies in the range of [-1,1]. If the correlation value for two variables x and y is negative, it means that when x increases y decreases and vice versa. If correlation is 0, it means the two variables x and y are not related whereas if correlation is positive, it means the two variables are linearly related to each other and have similar behaviour which means if one increases other also increases and vice versa. So, correlation between

actual and predicted variable should be close to 1 or -1 for any algorithm to be considered good.



(a) RMSE (b) MAE (c) Correlation

Figure 3.10: 10-fold cross validation of the different BEE models using different error metrics for power consumption. Legend: (—) CNN^7_{cov}, (- - -) CNN^9_{cov}, (—) CNN^7_{gaf}, (- - -) CNN^9_{gaf}, (—) CNN^7_{eig}, (- - -) CNN^9_{eig}

In this work, value of k has been taken as 10. So, the results for 10-fold cross validation using RMSE, MAE and correlation as the performance indicators are shown in Figure 3.10. It can be observed that CNN^7_{cov} has minimum RMSE and MAE and maximum correlation compared to the other CNN models. To further validate this conclusion, statistical analysis also was performed using the two sample t-test between the results of CNN^7_{cov} and the other CNN models i.e. CNN^7_{cov} with CNN^9_{cov}, CNN^7_{cov} with CNN^9_{eig} and so on. An analysis was conducted with the assumption that the populations have equal variances at the significance level of $\alpha = 0.05$. For the test, the null hypothesis was: the difference of population means is zero. Under this null hypothesis, the pooled t-test has been performed with the assumption of equal variance and hence t-statistics values, shown in Table 3.3, are obtained. It can be observed from the table that the t-stat values are greater than the t-critical values. Also, the p-values corresponding to each pair are less than the statistical significance level of $\alpha = 0.05$. Thus, the null hypothesis is rejected and the population means differ statistically significantly. Further, the mean of the RMSE and MAE values in case of CNN^7_{cov} is less than other CNN models and hence, the results obtained from CNN^7_{cov} are better than the other CNN models and this difference is statistically significant.

Table 3.3: Statistical analysis of BEE models with two sample t-test using RMSE and MAE of power consumption from 10-fold cross validation

	CNN^{9}_{qaf}	CNN^{7}_{qaf}	CNN^{9}_{eig}	CNN^{7}_{eig}	CNN^{9}_{cov}	CNN^{7}_{cov}
	Results using RMSE values from 10-fold cross validation					
Mean	3.26	3.76	12.46	13.62	1.84	1.39
Variance	0.41	0.37	0.56	0.15	0.07	0.05
Observations	10	10	10	10	10	10
Pooled Variance	0.26	0.23	0.34	0.11	0.07	-
Hypothetical mean difference	0	0	0	0	0	-
Degree of freedom	18	18	18	18	18	-
t-stat	8.19	10.97	42.28	81.61	3.91	-
$P(T \leq t)$ one tail	8.70×10^{-8}	1.05×10^{-9}	0	0	5.17×10^{-4}	-
t–critical one tail	1.734	1.734	1.734	1.734	1.734	-
	Results using MAE values from 10-fold cross validation					
Mean	2.29	2.54	8.38	9.65	1.06	0.88
Variance	0.57	0.26	0.45	0.24	0.01	0.02
Observations	10	10	10	10	10	10
Pooled Variance	0.33	0.15	0.26	0.14	0.02	-
Hypothetical mean difference	0	0	0	0	0	-
Degree of freedom	18	18	18	18	18	-
t-stat	5.49	9.41	32.79	51.61	3.03	-
$P(T \leq t)$ one tail	1.63×10^{-5}	1.13×10^{-8}	0	0	3.56×10^{-3}	-
t–critical one tail	1.734	1.734	1.734	1.734	1.734	-

3.3.5 Analysis of Results

A number of experiments were performed by training a number of CNN models with different number of layers (such as 5, 6, 7, 8, 9, 10 and 11) and varying the input feature descriptors, namely covariance, eigenvectors and GAF. From the results discussed above, it can be observed that different number of layers in the CNN model and different input feature descriptors have large impact on the performance of the BEE model. In brief, it can be concluded that the CNN models trained with covariance have performed really well as compared to CNN models trained with eigenvectors and GAF. The main reason for that is a lot of information is lost while calculating GAF and eigenvectors compared to covariance. Also, it can be concluded that as the number of layers increases the CNN models converge faster, for instance, the CNN models with 5, 7, 9 and 11 layers, when trained using the dataset preprocessed with covariance method, converged at approximately 420, 380, 330 and 290 epochs, respectively. Although, increasing the layers helps the models to converge earlier it increases the computational cost. Also, it is well known fact that the CNN models with more number of layers require more training data to achieve the same level of accuracy as CNN models with less number of layers. Therefore, it is important to find the minimum possible number of layers with acceptable performance. In this work, it has been observed that CNN model with 7 layers and trained with covariance feature descriptors, represented as CNN^{7}_{cov}, performed consistently better than other CNN models and

also it has been statistically validated in Subsection 3.3.4.

3.4 Comparative Analysis of BEE model and Other Techniques

To benchmark the results and to show the efficiency of the BEE model, the computed results are compared with five of the existing approaches. From discussions in section 3.3.3, 3.3.4 and 3.3.5, it was observed that CNN^7_{cov} performed better than other CNN models. Also, CNN models CNN^9_{cov} and CNN^9_{gaf} have comparable performance. So, in this section the results comparing outputs from this research CNN^7_{cov}, CNN^9_{cov} and CNN^9_{gaf} with five state-of-the-art techniques is presented.

i) To implement the multivariate model for power consumption estimation of EV the equation (3.12) proposed by Galvin [66] was used.

$$P = rV + sV^2 + tV^3 + uVA \tag{3.12}$$

where P, V and A represent the power demand, speed and acceleration, respectively and r, s, t and u are regression coefficients. The values for these coefficients for NissanSV as given in [66] are $r = 479.1$, $s = -18.93$, $t = 0.7876$ and $u = 1507$. According to the dataset used in this paper the variable V corresponds to v_{sp} and A corresponds to change in speed per unit time. So the equation (3.12) becomes

$$\begin{aligned} P(t) = 479.1v_{sp}(t) - 18.93v_{sp}(t)^2 + 0.7876v_{sp}(t)^3 \\ +1507v_{sp}(t)\Big(\frac{v_{sp}(t) - v_{sp}(t-1)}{(t) - (t-1)}\Big) \end{aligned} \tag{3.13}$$

where $P(t)$, $v_{sp}(t)$ represent the power demand and speed of the vehicle at time t, respectively.

ii) The model proposed by Yang et al. [44] also was implemented for comparison with the CNN models. Yang et al. [44] proposed the equations (3.14) and (3.15) for estimating the power consumption of EV when the motor runs in normal and regenerative mode,

respectively.

$$P = \frac{v}{\eta_{te}\eta_e}\left(\delta m\frac{dv}{dt} + mg(f+i) + \frac{\rho C_D A}{2}v^2\right) + P_{accessory} \tag{3.14}$$

$$P_{reg} = kv\eta_{te}\eta_m\left(\delta m\frac{dv}{dt} + mg(f+i) + \frac{\rho C_D A}{2}v^2\right) + P_{accessory} \tag{3.15}$$

where P is the power consumption, P_{reg} is power regenerated, v is the speed (corresponding to v_{sp} in $DS-I$ and $DS-II$ dataset), η_{te} is the transmission efficiency, η_e is driving efficiency, δ is the coefficient related to weight of EV, m is mass of vehicle, f is rolling resistance coefficient, i is the road grade (corresponding to r_{el} in $DS-I$ and $DS-II$ dataset), ρ is air density, C_D is the aerodynamic drag coefficient, A is the frontal area of vehicle, $P_{accessory}$ is the power consumed by accessories, k is the percentage of total energy during braking that can be recovered by the motor and η_m is the motor efficiency. Parameter k was defined using the following equation:

$$k = \begin{cases} 0.5 * \dfrac{v}{5} & v < 5\text{m/s} \\ 0.5 + 0.3\dfrac{v-5}{20} & v \geq 5\text{m/s} \end{cases} \tag{3.16}$$

For implementing the above model for Nissan Leaf 2013, values of m, C_D and A were used from Table 3.1. Other than these, values of δ, η_{te}, η_m, η_e, ρ, $P_{accessory}$ (assuming no AC or heater is running) and f given in [44] were 1.1, 0.9, 0.9, 0.8, 1.2, 150 and 0.015, respectively.

iii) A neural network model, as proposed by Diaz Alvarez et al. [75], with 14 inputs and 1 output but without hidden layer was trained using the mean and variance of three parameters as inputs. The parameters include speed, acceleration (further divided into positive and negative acceleration) and jerk (further partitioned into Starting Movement Jerk (SMJ), Cruising Track Jerk (CTJ), Starting Brake Jerk (SBJ) and Ending Brake Jerk (EBJ)). The neural network was trained with $DS-I_{tr}$ which is the 70% of data from $DS-I$ and validated using the rest 30%, denoted by $DS-I_{val}$.

iv) Similar to the above for comparison purpose a neural network, as developed by Felipe et al. [76] which is the extension of the neural network in [75], was trained with 137 inputs, 1 output, and no hidden layer. The input parameters include the mean and variance of road grade, accelerator pedal, brake pedal, speed, speed limits, acceleration, deceleration and

number of lanes etc along with the jerk parameters used in [75]. This neural network was also trained with 70% of data from $DS-I$ and validated with the rest.

v) The MLR (Multiple Linear Regression) model proposed by De Cauwer et al. [65] was implemented to estimate the energy consumption of EV for a trip divided into number of small segments using the following equation:

$$\begin{aligned}\triangle E = \sum_{\text{segments } j}^{trip} \Big[& B_1 \triangle s_j + B_2 \sum_{i}^{n} (v_{EV_i} + v_{wi})^2 \triangle s_j + B_3 (CMF_j^+) \triangle s_j \\ & + B_4 (CMF_j^-) \triangle s_j + B_5 \triangle Hpos_j + B_6 \triangle Hneg_j + B_7 Aux_{Tj} \triangle t_j + \varepsilon \Big]\end{aligned} \tag{3.17}$$

with:

$$CMF_j = \frac{\sum_{i=2}^{n} \mid v_{EV_i}^2 - v_{EV_{i-1}}^2 \mid}{\triangle s}$$

where B_i, $\triangle E$, v_{EV_i}, v_{wi}, $\triangle s$, $\triangle s_i$, Aux_T, $\triangle t$, $\triangle Hpos_j$, $\triangle Hneg_j$, ε, n are regression coefficients, energy, vehicle speed at time t_i, wind speed at time t_i, distance, driven distance between t_{i-1} and t_i, temperature scaling, time, positive elevation changes, negative elevation changes, error term and number of data points in segment j, respectively. For implementing the above model the drive cycles were divided into small segments each of 10 sec duration. Then for each segment equation (3.17) was applied by using the values from dataset $DS-I$ and $DS-II$. For instance v_{sp} for v_{EV}, average v_{sp} of segment j multiplied by time for $\triangle s_j$ etc. The values of regression coefficients B_1 to B_7 for Nissan Leaf was provided in [65], so those values were used for comparison with the BEE models.

The values of different performance metrics of the above discussed five techniques and the BEE models CNN_{cov}^7, CNN_{cov}^9 and CNN_{gaf}^9 are presented in Table 3.4 and it can be observed that CNN_{cov}^7 outperforms the other existing approaches with a lowest mean E_{dev} of $5.21\% \pm 2.95$ and $5.09\% \pm 1.31$ on dataset $DS-I_{val}$ and $DS-II$, respectively. These results have also been validated by other metrics such as Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Correlation (Corr). Also, it can be observed that the BEE models CNN_{cov}^7 and CNN_{cov}^9 have lowest RMSE (i.e. 1.39 and 1.84 on $DS-I_{val}$ and 1.35 and 1.46 on $DS-II$) and MAE (i.e. 0.88 and 1.06 on $DS-I_{val}$ and 0.76 and 0.85 on $DS-II$) and highest Corr (i.e. 0.995 and 0.993 on $DS-I_{val}$ and 0.997 and 0.996 on $DS-II$) values, as compared to the existing techniques. This shows that CNN_{cov}^9 is the second best model after CNN_{cov}^7, in terms of RMSE, MAE and Corr. Values of RMSE, MAE and Corr can not be calculated for [75, 76] as the techniques presented in these do not give real-time power/energy consumption as output and provide only a single value of total energy consumption for the trip. It can be observed that all the approaches performed better on $DS-II$ than on $DS-I_{val}$. The main

Table 3.4: State-of-the-art comparison using different performance metrics

Approach	$DS - I_{val}$				$DS - II$				Average Prediction Time / drive cycle (in sec)
	Mean E_{dev}	RMSE	MAE	Corr	Mean E_{dev}	RMSE	MAE	Corr	
De Cauwer et al. [65]	7.25	4.22	1.70	0.953	6.09	1.85	0.95	0.982	1.58×10^{-3}
Yang et al. [44]	8.78	6.19	3.13	0.935	8.13	3.45	2.36	0.977	1.97×10^{-3}
Galvin [66]	13.63	8.54	3.87	0.763	11.56	2.33	1.11	0.981	$\mathbf{3.47 \times 10^{-4}}$
Diaz Alvarez et al. [75]	12.37	NA	NA	NA	10.21	NA	NA	NA	1.14×10^{-2}
Felipe et al. [76]	7.41	NA	NA	NA	7.34	NA	NA	NA	3.06×10^{-2}
The BEE Models*									
CNN^7_{cov}	**5.21**	**1.39**	**0.88**	**0.995**	**5.09**	**1.35**	**0.76**	**0.997**	1.76
CNN^9_{cov}	9.43	1.84	1.06	0.993	8.86	1.46	0.85	0.996	2.26
CNN^9_{gaf}	13.38	3.26	2.29	0.972	12.07	3.01	1.99	0.978	2.28
NA represent not applicable and bold values are the best ones									

* The models developed in this chapter

reason for this is dataset $DS - II$ had readings on constant road grade of 0% i.e. no change in road elevation whereas $DS - I_{val}$ had readings with road grade varying from -20% to 20%. Results for another metric namely, the average prediction time per drive cycle is also given in Table 3.4. The average prediction time per drive cycle is the time, the trained model takes to predict the output for given driven cycle and does not include the training time of the model. It has been calculated on a system with Intel i5 Processor, 8GB RAM on a torch-lua platform. It can be seen that the BEE models CNN^7_{cov}, CNN^9_{cov} and CNN^9_{gaf} take more inference time compared to existing techniques. This is due to the fact that the BEE models have more layers in the architecture which increases the amount of computation required. This is also evident from the prediction time of CNN^7_{cov} and CNN^9_{cov}, where CNN^9_{cov} takes 2.26 sec compared to CNN^7_{cov} which takes 1.76 sec. CNN^7_{cov} takes less time to predict the output because it has fewer layers. However, the current aim is to compare the accuracies of the architecture with previous techniques. Once the model is verified in terms of accuracy, it can be converted to a TensorFlow Lite format to be suitable for execution on Google Coral boards or Odroid boards with Movidius sticks, hence providing a real time performance with a very low inference time.

A comparison of energy consumption, estimated using the BEE models CNN^7_{cov}, CNN^9_{cov} and CNN^9_{gaf} and state-of-the-art approaches [44, 65, 66] are presented in Figure 3.12. The energy consumption was calculated by integrating the power over the time period. As [75, 76] do not provide real-time power/energy consumption as output so, it was not possible to plot them. In Figure 3.12, each column represents the different road grade profile and each row represents the different drive cycle i.e., from top to bottom, rows represent UDDS, SFTP, HWFET and NEDC drive cycles, respectively and from left to right, columns represent Grade Profile 1 (constant grade at 0% i.e., no change in elevation), Grade Profile 2 (varies from -2% to 2%) and Grade Profile 3 (varies from -20% to 15%), as shown in Figure 3.11. It can be clearly observed from the Figure 3.12 that in most cases CNN^7_{cov} has performed better than all of the

existing approaches [44, 65, 66] and the two other CNN models CNN^9_{cov} and CNN^9_{gaf}, in terms of accuracy of estimates. The difference in estimates can be seen more clearly when road grade varies within a larger range such as in Grade Profile 3 i.e., in the rightmost column of Figure 3.12. Also, it can be concluded that the estimates given by BEE models follow the same behaviour or trend as the actual energy consumption whereas the previous techniques display large deviation from the actual value.

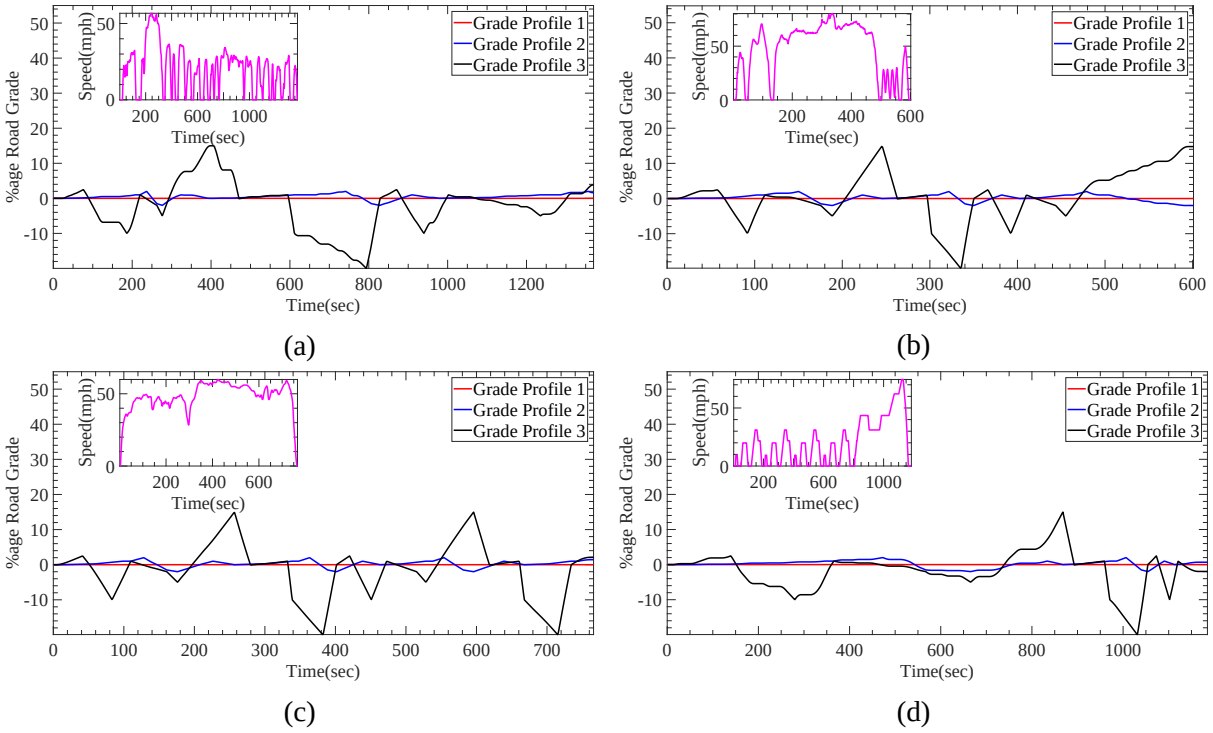
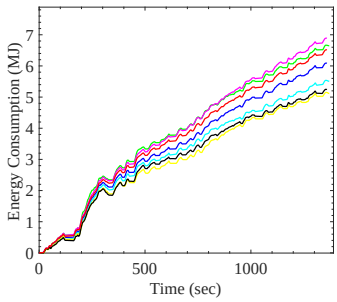
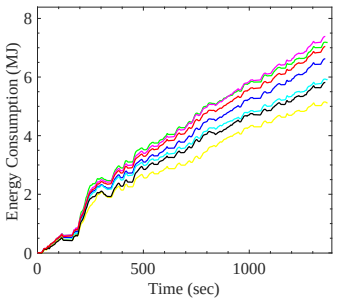


Figure 3.11: Grade Profiles for different drive cycles (shown in sub-axis), (a) UDDS, (b) SFTP, (c) HWFET and (d) NEDC

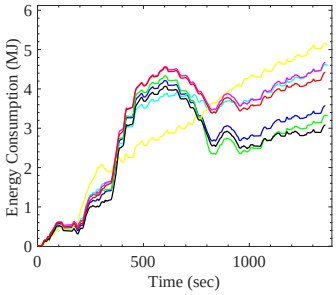
As Galvin [66] did not consider the effect of road grade so it is justified for his model to deviate from actual energy consumption when road grade effect gets introduced. Yang et al. [44] considered the effect of road grade but they tested their model only for small tilt angles i.e. $0°$, $1°$, $2°$ and $3°$ so when the tilt angle changes with large values as in Grade Profile 3, where road grade varies from -20% (i.e. $-11.30°$) to 15% (i.e. $8.53°$), their model fails to estimate the actual energy consumption accurately. The MLR model developed by De Cauwer et al. [65] has performed really well and in some cases even performed better than the CNN^7_{cov} but it also fails to estimate the energy consumption to an acceptable level of statistical confidence when road grade changes with high values. The main reason for this is the non-linear relationship of the influencing parameters which the MLR model was not able to estimate with comparable statistical confidence as the CNN^7_{cov}. The NN architecture presented in [75, 76] have no hidden layer and has one input layer with 14 and 137 inputs, respectively and 1 output corresponding to the inputs. As the networks were shallow, they also were not able to reasonably represent



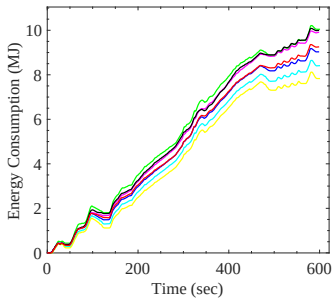
(a) Results for UDDS with grade profile 1



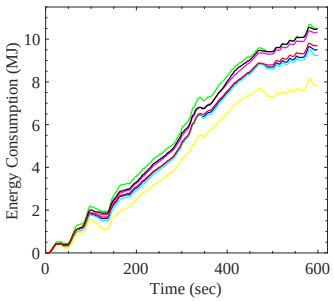
(b) Results for UDDS with grade profile 2



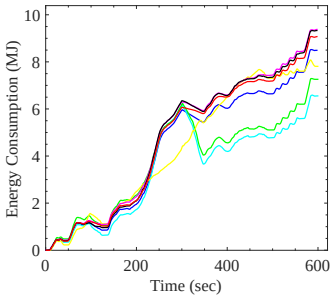
(c) Results for UDDS with grade profile 3



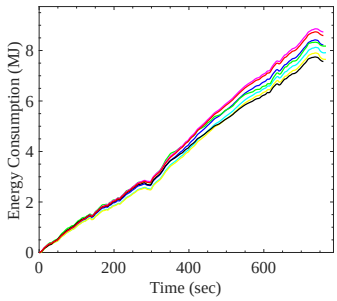
(d) Results for SFTP with grade profile 1



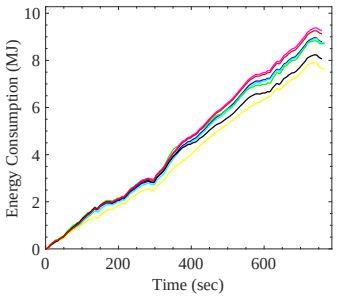
(e) Results for SFTP with grade profile 2



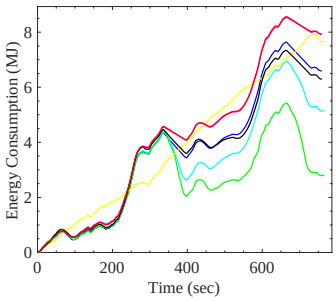
(f) Results for SFTP with grade profile 3



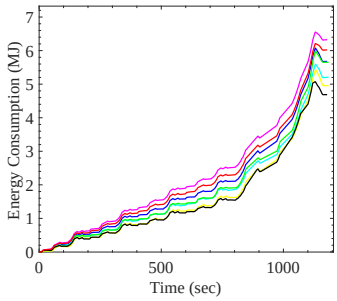
(g) Results for HWFET with grade profile 1



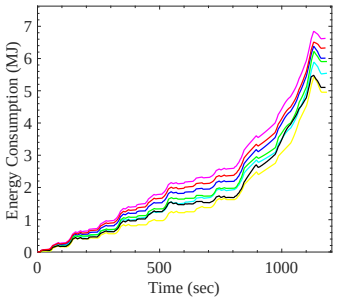
(h) Results for HWFET with grade profile 2



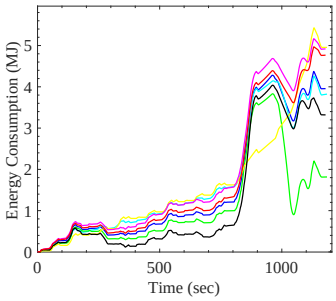
(i) Results for HWFET with grade profile 3



(j) Results for NEDC with grade profile 1



(k) Results for NEDC with grade profile 2



(l) Results for NEDC with grade profile 3

Figure 3.12: Energy consumption estimation comparison of BEE models and existing techniques for different driving cycles and road grade profiles. Legend: (—) Actual / Target, The existing approaches (—) De Cauwer et al. [65], (—) Yang et al. [44] and (—) Galvin [66], The BEE models* (—) CNN^{7}_{cov}, (—) CNN^{9}_{cov} and (—) CNN^{9}_{gaf}

* The models developed in this chapter

the non-linear relationship between the influencing factors. Also, the NN architectures provide only one output of total energy consumption over the whole trip so they can not be used to provide real-time information to the drivers.

From the above discussions, a number of observations can be concluded which are summarized as follow:

(i) The results show that although the CNN model with more layers i.e., CNN^9 converge faster than a model with fewer layers i.e., CNN^7, an increase in number of layers does not necessarily increase the estimation accuracy.

(ii) Road gradient is an important parameter that effects the energy consumption of EV greatly. This is why it can be seen that with large road grade changes most of the existing techniques fail to accurately estimate the energy consumption.

(iii) CNN models trained with covariance feature descriptors i.e. CNN^7_{cov} and CNN^9_{cov} give very good results than CNN models trained with GAF or Eigen feature descriptors so the choice of input features also affect the performance of CNN architectures.

3.5 Summary

Deep Convolutional Neural Networks (D-CNN) based solution has been developed for estimation of energy consumption of EVs considering three external parameters namely, road elevation, tractive effort and speed of the vehicle. Unlike previous methods that require either manufacturer data, which is not readily available, or real-world data, which require special sensors to be deployed on EVs, the BEE model requires only three parameters which can easily be obtained. A number of CNN models with different architectures were trained using simulated data after preprocessing, in which the simulated time series data was converted to images. The data was generated from a simulation model of Nissan Leaf 2013, developed in FASTSim. The CNN models were tested using the experimental data obtained from Argonne National Laboratory of US. It has been observed that one of the CNN models with seven layers represented with CNN^7_{cov} performed really well with average percentage energy consumption deviation of $5.21\% \pm 2.95$ and $5.09\% \pm 1.31$ on dataset $DS - I_{val}$ and $DS - II$, respectively. In the next chapter, an improved model which also considers the effect of other parameters such as environmental temperature, traffic and auxiliary loads etc. will be presented.

Chapter 4

Improved Energy Estimation (IEE) Model [1]

An Improved Energy Estimation (IEE) model using Convolutional Neural Network (CNN) and Bagged Decision Tree (BDT) has been developed to provide a real-time estimates of EVs energy consumption with an acceptable level accuracy. The CNN used is multi-channel i.e. there are multiple parallel branches where each branch extracts important features from individual input parameter and then the extracted features are combined to predict the energy consumption. The IEE model also uses a BDT, which consist of multiple decision trees ensembled together using weighted average ensembling. The BDT is used as a fine tuner to improve the initial estimate given by the CNN by decreasing the prediction error further. The IEE model is the extension of Basic Energy Estimation (BEE) model, discussed in the previous chapter, and does not require any vehicle specific parameters for it to work. It takes seven input parameters namely, vehicle speed, vehicle acceleration, road elevation, wind speed, auxiliary loads, environmental temperature and initial State of Charge (SOC) of battery. These parameters are easily available, for instance, vehicle speed is easily available from vehicle, acceleration can be computed from the speed, Geographic Information System (GIS) can be used to obtain road elevation, freely available weather API's (such as API's from OpenWeatherMap [46]) can be used to obtain wind speed and environmental temperature etc. In comparison to the BEE model discussed in Chapter 3, the IEE model provides better real-time energy consumption estimates in terms of accuracy, as it considers the effect of almost all the influencing factors namely, vehicle speed, acceleration, wind speed, auxiliary loads, battery's SOC, environmental temperature and road elevation.

This chapter consists of four sections. Section 4.1, describes the architecture of the IEE model. Section 4.2, discusses the experimental setup for the IEE model i.e. the datasets used and training parameters etc. The IEE model was compared with multiple other existing techniques and a comparison of the results is discussed and analysed in Section 4.3. Finally, Section 4.4 summarizes the results and concludes the chapter. Note that, unless otherwise stated auxiliary

[1] The content of this chapter is published as "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 Computing*, vol. 25, no. 3, pp. 2399–2416, 2021. DOI: https://doi.org/10.1007/s00500-020-05310-y (SCI Impact Factor: 3.050)

load is in W, temperature in °C, time in seconds, speed in m/s, vehicle weight in kg, power in KW and energy in MJ.

4.1 Architecture of IEE model

There are number of factors such as environmental temperature, wind speed, road elevation, the battery's SOC etc which influence the energy consumption of an EV. These factors vary significantly in real life and have a non-linear relationship among them. Therefore, to effectively represent their non-linear relationship a hybrid approach has been adopted, which can provide more accurate power / energy consumption estimates of an EV under different conditions. Figure 4.1, shows the IEE model's architecture. There are four main computational modules of the IEE model namely, Power Consumption Estimation (PCE), Re-sampler, Fine Tuner and State of Charge Calculator. The IEE model takes seven inputs namely, vehicle speed, vehicle acceleration, environment temperature, wind speed, auxiliary loads, road elevation and battery's initial SOC. As discussed in Section 4.2.1, these inputs are short partitioned time series of 10 Hz frequency each of 10 sec duration i.e. each partition has 100 readings for 10 sec duration. The environmental temperature and battery's initial SOC does not change much in the duration of 10 sec so they have been considered as constant for a particular partition. The inputs are provided simultaneously to the PCE and Re-sampler module. The PCE module consists of a CNN model, which uses the seven inputs to provide an estimated power consumption for the EV. The power consumption estimate given by the PCE module is of 1 Hz frequency. The Re-sampler module takes the seven inputs and up/down sample them to match the frequency of inputs with output from PCE module i.e. 1 Hz. Then, the re-sampled data inputs along with estimated

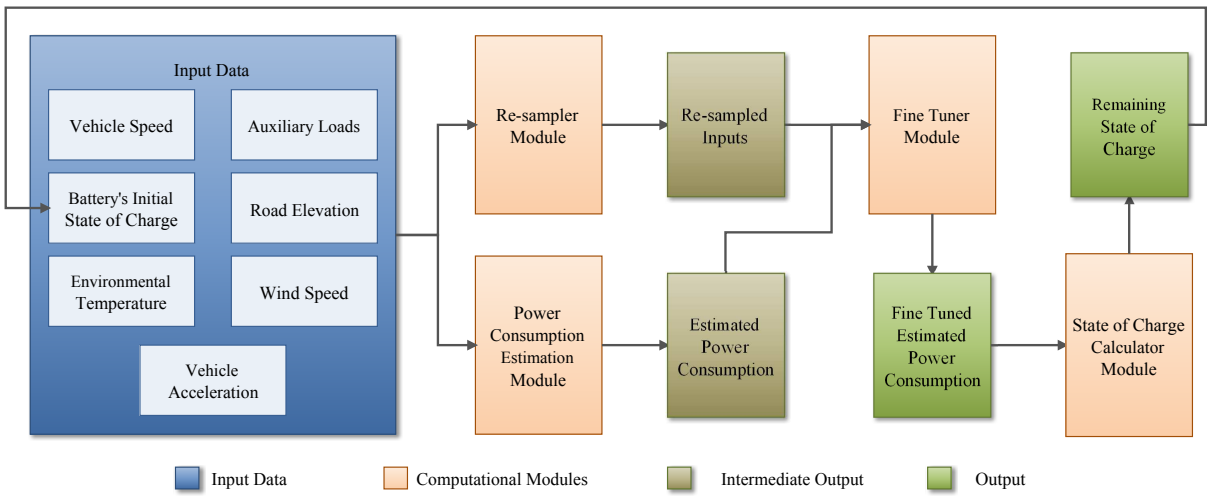


Figure 4.1: Architecture of IEE model

power consumption, are further passed through a Fine Tuner module which uses BDT to fine tune the estimate of power consumption by reducing the error. The fine tuned estimated power consumption is then used by the State of Charge Calculator module. This calculates the remaining state of charge of the battery based on the power consumed by the EV. This calculated remaining SOC is used as the initial battery SOC for the next partition of 10 sec. The following sub-sections discuss in detail the workings of these four computational modules of the IEE architecture.

4.1.1 Power Consumption Estimation (PCE) Module

The Power Consumption Estimation (PCE) Module is responsible for estimating the power consumption of EV under different environmental and road conditions. For this a multi-channel CNN, as shown in Figure 4.2, has been developed. The multi-channel CNN is inspired from the network architecture used in [126], originally developed for hand gesture classification from time series pose data. In this work, the IEE architecture takes seven input time series namely, vehicle's speed (v_{sp}), road elevation (r_{el}), vehicle's acceleration (v_{acc}), auxiliary loads (a_{ld}), wind speed (w_{sp}), initial state of charge of battery (b_{soc}) and environmental temperature (e_{temp}). As discussed in Section 4.2.1, all of these time series (say x) in the dataset were recorded at 10 Hz frequency and were partitioned into n small time series (say $x^i, i = 1, 2, ..., n$) each of 10 seconds duration. These small time series each of 10 sec duration became the input for the network. It has been observed that out of these seven input parameters the first five parameters vary a lot but the environmental temperature (e_{temp}) and battery's initial SOC (b_{soc}) does not change during the interval of 10 seconds. Due to this, these two parameters have been considered as constant for each partition and no feature extraction has been performed for these two parameters. Each of the other five parameters were passed to a separate feature extraction module, each of them has four separate branches to extract features. Each feature extraction module has one residual branch and three similar convolutional branches.

The residual branches make gradient backpropagation better during the training and hence the network optimizes easily which ultimately increases the accuracy of the network [127]. The residual branch is acting like an identity function but instead of giving the same output as the input three average pooling layers were used. The average pooling layers downsample the data by taking the average of the input data from a particular region. The pooling layers make the CNN locally invariant i.e. the CNN can extract the same features from the input regardless of rotation, scaling or shifting of features [104]. In this way, the pooling layers not only reduce the network scale, but also extract the important features from the input. This also helps in avoiding overfitting of the network. The main feature extraction is carried out by the three

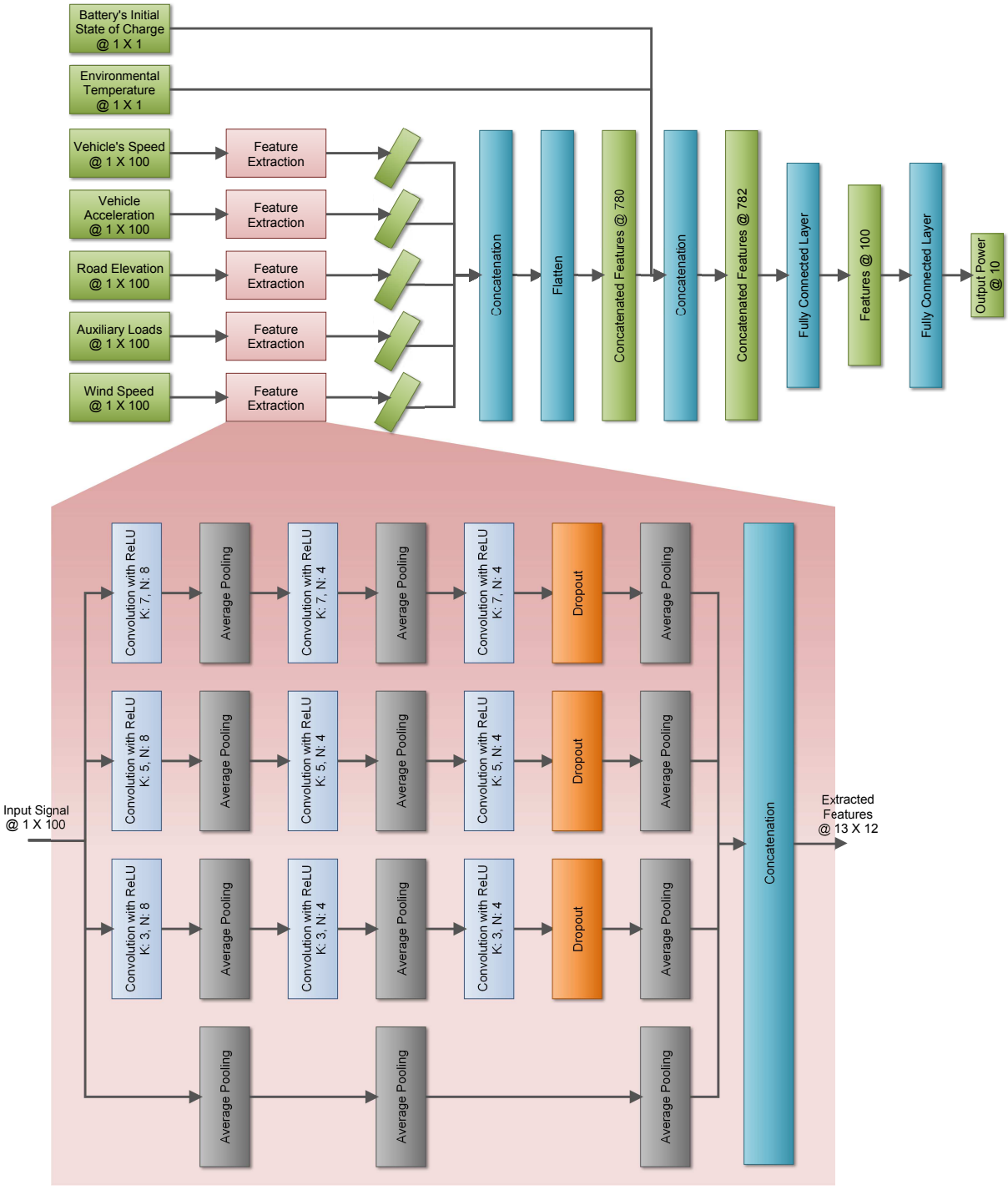


Figure 4.2: Architecture of Power Consumption Estimation (PCE) Module. (Numbers after K, N and @ represent the kernel size, number of kernels and the dimension of vector, respectively)

convolutional branches which have similar architecture, as discussed below.

Each convolutional branch has three convolutional layers followed by an average pooling layer except the last convolution layer which is followed by a dropout layer and then a pooling layer. Convolutional layers are different from traditional fully connected layers as they apply convo-

lutional filters on the input. The convolutional filters extract the local features by focusing on a particular region of the input. One convolutional layer can have a hundred of such filters and extract hundreds of features from the input. This helps the convolutional layers to learn the complex patterns. In a single branch, the convolutional layers differ from each other in terms of the number of kernels used, for instance, in each convolutional branch the first convolutional layer has 8 kernels whereas the second and third convolutional layer has 4 kernels. Convolutional branches differ from each other in terms of size of kernels i.e. one convolutional branch has kernels of size 3, second has kernels of size 5 and third has kernels of size 7. Branches with different kernel size help the network to extract and learn the features based on different time resolutions. Each convolution layer has padding of n, which depend on kernel size, as given by the equation $n = (kernel\_size - 1)/2$. ReLU has been used as an activation function for each convolutional layer and can be defined as $ReLU(x) = max(0, x)$. The activation function in CNN has the two main advantages. First, the activation function convert the output to a scaled range which helps fast training of the network. Secondly, the combination of activation function help the network learn more complex non-linear patterns from the input. Each convolutional branch has a dropout layer as a regularizer [128]. During training, the dropout layers make some outputs from the previous layer randomly ignored. This makes the training process noisy and forces the layers to co-adapt for mistakes made by prior layers which ultimately make the network more robust.

The output features from the convolutional and residual branches are concatenated at the end which results in the output features of size 13×12 extracted by the feature extraction module. The extracted features from each feature extraction module are further concatenated and then flattened in the consecutive layers. These flattened feature vectors of size 780 and the two parameters of environment temperature (e_{temp}) and the battery's initial SOC (b_{soc}) are concatenated to form the final feature vector of size 782 which is then passed through two successive fully connected layers to give the final output of estimated power consumption of size 10 for the 10 second interval.

4.1.2 Re-sampler Module

The Re-sampler module up-sample or down-sample the input parameters to match the frequency of output generated by PCE Module. As discussed in the PCE module, the estimated power consumption output generated is of 1 Hz frequency i.e. one data point for each second, so it is necessary to re-sample the input parameters to match the frequency of 1 Hz before giving them as input to the Fine Tuner module. The input parameters namely, temperature and the battery's initial SOC, as mentioned in PCE module are considered as constant for each 10 sec-

ond interval. Due to this, the input parameters other than environment temperature and battery's initial SOC are down-sampled from 10 Hz to 1 Hz and the input parameters of temperature and battery's initial SOC are up-sampled to 1 Hz by repeating the same value 10 times.

4.1.3 Fine Tuner Module

The fine tuner module takes eight input parameters namely, vehicle's speed, road elevation, vehicle's acceleration, auxiliary loads, wind speed, environmental temperature, battery's initial SOC and estimated power consumption. The first seven of these parameters are re-sampled by the Re-sampler module and the last one is estimated by the PCE module. The main purpose of the fine tuner module is to fine tune the estimated power consumption based on the other input parameters by reducing the error in the prediction. Multiple experiments were performed with different regression models and it was found that Bagged Decision Trees (BDT) performed better than other regression models. The BDT is a collection of multiple decision trees ensembled together using bagging. Bagging or bootstrap aggregation is an ensemble learning technique proposed by Breiman [129]. The main objective of bagging is to reduce the variance of learners, in this case the decision trees. This is achieved by combining the output of multiple learners through averaging which ultimately help to achieve high prediction accuracy. In BDT, each learner is a separate decision tree and each decision tree is trained separately on a subset of the training dataset. For this, the training dataset is divided into multiple subsets using the bootstrap resampling. In bootstrap resampling, samples are selected from the training dataset with replacement. The output from all the separate decision trees are then averaged to obtain the final predictions which is, in this case, the fine tuned estimated power consumption. In this work, the number of trees and other parameters of the BDT are optimized by minimizing the cross validation error. The optimized model contains 10 weak tree learners, ensembled together in the BDT with minimum prediction accuracy.

4.1.4 State of Charge Calculator Module

The fine tuned estimated power consumption obtained from the Fine Tuner module is used by the State of Charge Calculator Module to calculate the remaining state of charge of the battery. For this, the following equation, given in [130], was used:

$$SOC_t = SOC_0 - \left(\frac{\int_0^t P_{est} dt}{E_{cap}} \times 100 \right) \tag{4.1}$$

where SOC_0, SOC_t represent the battery's initial state of charge and state of charge at time t.

P_{est} is the fine tuned estimated power consumption for the time interval and E_{cap} represents the battery's rated energy capacity. In the current case, as each of the small partitioned time series is of 10 sec duration, value of t will be 10. Therefore, the system requires the initial SOC only for the first partitioned time series and after that it can calculate the SOC based on the energy consumed by the vehicle and the calculated SOC can be used as initial SOC for the next time series partition.

4.2 Experimental Setup

In this section, the details of the dataset, hyperparameters and other information related to the training of the IEE model is discussed.

4.2.1 Datasets Used

For training, testing and validating the IEE model data set from two different sources was used. The first source was Downloadable Dynamometer Database (D^3) [56] which contains data obtained by performing a number of dynamometer tests at the Argonne National Laboratory (ANL) of Advanced Powertrain Research Facility (APRF) on a number of electric vehicles. These tests were conducted for a number of drive cycles at 0% road elevation under different environmental conditions. This is the same dataset used for testing the BEE model presented in the previous chapter.

As the data available at D^3 was very small and was not sufficient for training, testing and validating the model, a large dataset was obtained from a Nissan Leaf's simulated model provided in a simulation tool named FASTSim (Future Automotive Systems Technology Simulator) [50]. The simulation model used Nissan Leaf's vehicle specific parameters which can be obtained from [116, 117]. The dataset for other electric vehicles also can be obtained by developing similar simulated models for other electric vehicles subject to the availability of vehicle data from manufacturer such as efficiency curve of motor, internal resistance of the battery etc. The dataset was generated using the simulated model for 5 different temperatures (from $-5°C$ to $35°C$ at an interval of $10°C$), 10 road elevation profiles (road grade varies from -20% to 20%), 4 different initial state of charge of battery (from 30% to 90% with intervals of 20%), 40 drive cycles (such as UDDS (Urban Dynamometer Driving Schedule) and SFTP (Supplemental Federal Test Procedures) etc) and 8 wind speed profiles. Wind speed was categorized into 13 different groups according to the Beaufort scale [131], namely, Calm (< 0.5 m/s), Light Air (0.5 - 1.5 m/s), Light Breeze (1.6 - 3.3 m/s), Gentle Breeze (3.4 - 5.5 m/s), Moderate Breeze (5.5 -

7.9 m/s), Fresh Breeze (8 - 10.7 m/s), Strong Breeze (10.8 - 13.8 m/s), Near Gale (13.9 - 17.1 m/s), Gale (17.2 - 20.7 m/s), Strong Gale (20.8 - 24.4 m/s), Storm (24.5 - 28.4 m/s), Violent Storm (28.5 - 32.6 m/s) and Hurricane ($\geq$ 32.7 m/s). Out of these 13 categories the first 8 were used for the data generation from the simulated model because the remaining categories are not suitable conditions for driving. It is to be noted that more data can be generated by varying the environmental and road conditions and using new drive cycles.

From now onwards in this chapter, the dataset obtained using the simulated model of FASTSim and the dataset downloaded from D^3 will be denoted as $DS-I$ and $DS-II$, respectively. Both datasets have time series data which is recorded at a frequency of 10 Hz i.e. for each second there are 10 readings. In order to train, validate and test the IEE model the datasets were partitioned into a number of segments each of 10 seconds interval. So, in total the dataset $DS-I$ and $DS-II$ were partitioned into approximately 17 lacs and 3500 partitions, respectively. 70% of the partitions of dataset $DS-I$ were used for training the IEE model and the remaining, i.e. 30%, were used for validation. Henceforth in this chapter, the training and validation dataset will be represented by $DS-I_{tr}$ and $DS-I_{val}$, respectively. The testing of the IEE model was carried out using dataset $DS-II$.

4.2.2 Data Preprocessing

As discussed in Section 4.2.1, both datasets have data of multiple parameters recorded at 10 Hz frequency. For experimental purposes, the time series data from the datasets were divided into smaller partitions each of 10 sec interval i.e. each partition has 100 readings. Time series data for each parameter (say z) from the dataset was normalized, using the Eq. (4.2), into the range of [0,1] before using it for training, validation or testing the IEE model.

$$\hat{z^i} = \frac{z^i - \min(z)}{\max(z) - \min(z)} \tag{4.2}$$

In above equation, $\hat{z^i}$, z^i, $\min(z)$ and $\max(z)$ represent the i^{th} normalized partition of time series z, i^{th} partition of time series z, minimum and maximum values of time series z, respectively.

4.2.3 Training of IEE Model

In the IEE model, there are mainly two modules, namely, PCE Module and Fine Tuner Module. The PCE module was implemented in Python using PyTorch APIs and the Fine Tuner module

was implemented in MATLAB 2019a. PyTorch packages, *torch.nn* and *torch.optim*, were used to define the multi-channel CNN architecture for PCE module and loss functions used for learning. It also provides a number of APIs for training and testing the created model. For training the BDT of Fine Tuner Module an addon, named Regression Learner App, provided in MATLAB was used. This can be used for training a number of different regression models in MATLAB. The implementation code for the IEE model along with the sample data has been provided on GitHub (*https://github.com/shatrughanmodi/CNN-BDT*). After training the PCE and Fine Tuner modules separately, both modules were integrated by calling MATLAB code from Python. The training of modules was done on a system with 8 GB RAM and Intel i5 Processor. While training the modules, the feedback loop, in which the remaining SOC calculated after each 10 sec interval was fed to the initial SOC of the next interval, was not used, but the feedback loop was used while testing the IEE model.

First the PCE module was trained using the 70% of time series data from the dataset $DS-I$. For training the PCE module, the weights of all the convolutional and fully connected layers were initialized using the Xavier initialization [132], which initializes the layer's weight from a random uniform distribution with limits of $\left[-\sqrt{\frac{6}{fan_{in}+fan_{out}}}, \sqrt{\frac{6}{fan_{in}+fan_{out}}}\right]$, where fan_{in} and fan_{out} are the number of input connections to the layer and number of output connections from the layer, respectively. The PCE module was trained with batch size of 64 for 3000 epochs using Adam optimization algorithm [133]. The Adam optimizer has the combined advantages of two stochastic gradient descent algorithms i.e. AdaGrad and RMSProp. The Adam Optimizer uses the delta learning rule to update the weights of the network. It basically computes an exponential running average of the gradients and square of the gradients. The decay rate of these running averages are controlled by the parameters β_1 and β_2 which were initialized to 0.9 and 0.999, respectively. The initial learning rate α was set to 0.001 and to avoid division by zero during training epsilon ϵ was set to 10^{-8}. These values of β_1, β_2, α, and ϵ were selected using hyperparameter tuning which showed the optimal results with the above values. The Mean Square Error (MSE) was used as the loss function to minimize the MSE between the actual and estimated power consumption. To avoid overfitting, dropout layers were used as regularizers in each convolutional branch of the feature extraction module. A number of experiments were performed by varying the drop rate p and it was found that increasing the drop rate after a certain threshold (in this case 0.2) does not reduce the testing error. After training the PCE module, the Fine Tuner Module was trained using the output obtained from PCE module and re-sampled input data obtained from the Re-sampler module.

The bagged decision tree of the Fine Tuner Module was developed by ensembling the multiple decision trees. The number of trees and other parameters were optimized by using the Bayesian optimization which tries to minimize the cross validation error. The model, optimized with

Bayesian optimization, has 10 weak tree learners which were ensembled to form a bagged decision tree with minimum prediction error.

4.3 Experimental Results and Analysis

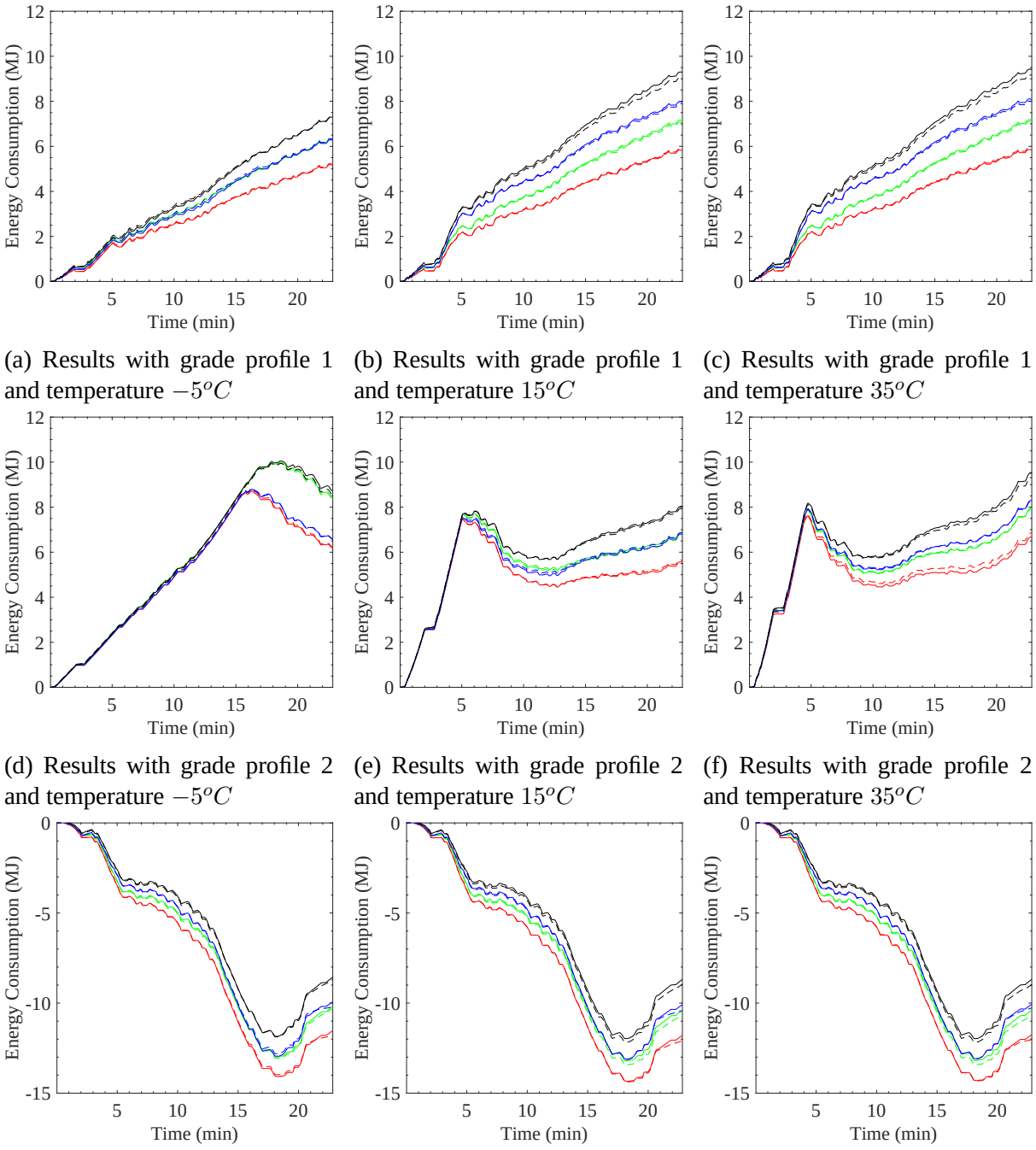
In this section, the results obtained using IEE model are discussed. Figure 4.3 and 4.4, show the energy consumption prediction comparisons for the IEE model with the actual energy consumption of EV under different conditions for the UDDS drive cycle with initial battery's SOC level at 30% and 70%, respectively. The different profiles of road grade, air speed, auxiliary load and vehicle speed, which were used to obtain the results shown in above mentioned figures, are given in Table 4.1.

Table 4.1: Different input parameter profiles for results shown in Figure 4.3 and 4.4

Parameter Name	Profile	Minimum	Maximum	Mean	Standard Deviation
Vehicle Speed (m/s)	UDDS	0	25.347	8.747	6.576
Auxiliary Load (W)	Profile 1	0	0	0	0
	Profile 2	952.941	1124.705	977.649	41.605
Air Speed (m/s)	Profile 1	0.084	0.214	0.150	0.019
	Profile 2	11.305	13.187	12.299	0.353
Road Grade (%)	Profile 1	0	0	0	0
	Profile 2	-18.765	17.696	1.198	9.107
	Profile 3	-17.953	10.262	-8.064	9.883

There are number of observations that can be drawn from the Figures 4.3 and 4.4. From a single sub-figure of these figures, it can be observed how the auxiliary loads and air speed influence the energy consumption of EV. For instance, observe Figure 4.3c, energy consumption for four combinations of air speed and auxiliary loads have been presented each with different colors. It can be seen that when the wind flows at a higher speed, as in air speed profile 2, and also auxiliary loads have been applied, as in auxiliary load profile 2, the energy consumption is highest compared to the energy consumption with other combinations of air speed and auxiliary loads. There is approximately a 50% increase in energy consumption under air speed profile 2 and auxiliary load profile 2 compared to under air speed profile 1 and auxiliary load profile 1 (i.e. no auxiliary load). So, a significant amount of energy is consumed to overcome the aerodynamic drag and fulfill the demand of auxiliary loads as expected.

The effect of road grade profile can be seen clearly by comparing the energy consumption within a column of the Figures 4.3 and 4.4, where all other parameters are the same but only road grade varies. For instance, consider the middle column of Figure 4.4, the energy consump-



(a) Results with grade profile 1 and temperature -5^oC

(b) Results with grade profile 1 and temperature 15^oC

(c) Results with grade profile 1 and temperature 35^oC

(d) Results with grade profile 2 and temperature -5^oC

(e) Results with grade profile 2 and temperature 15^oC

(f) Results with grade profile 2 and temperature 35^oC

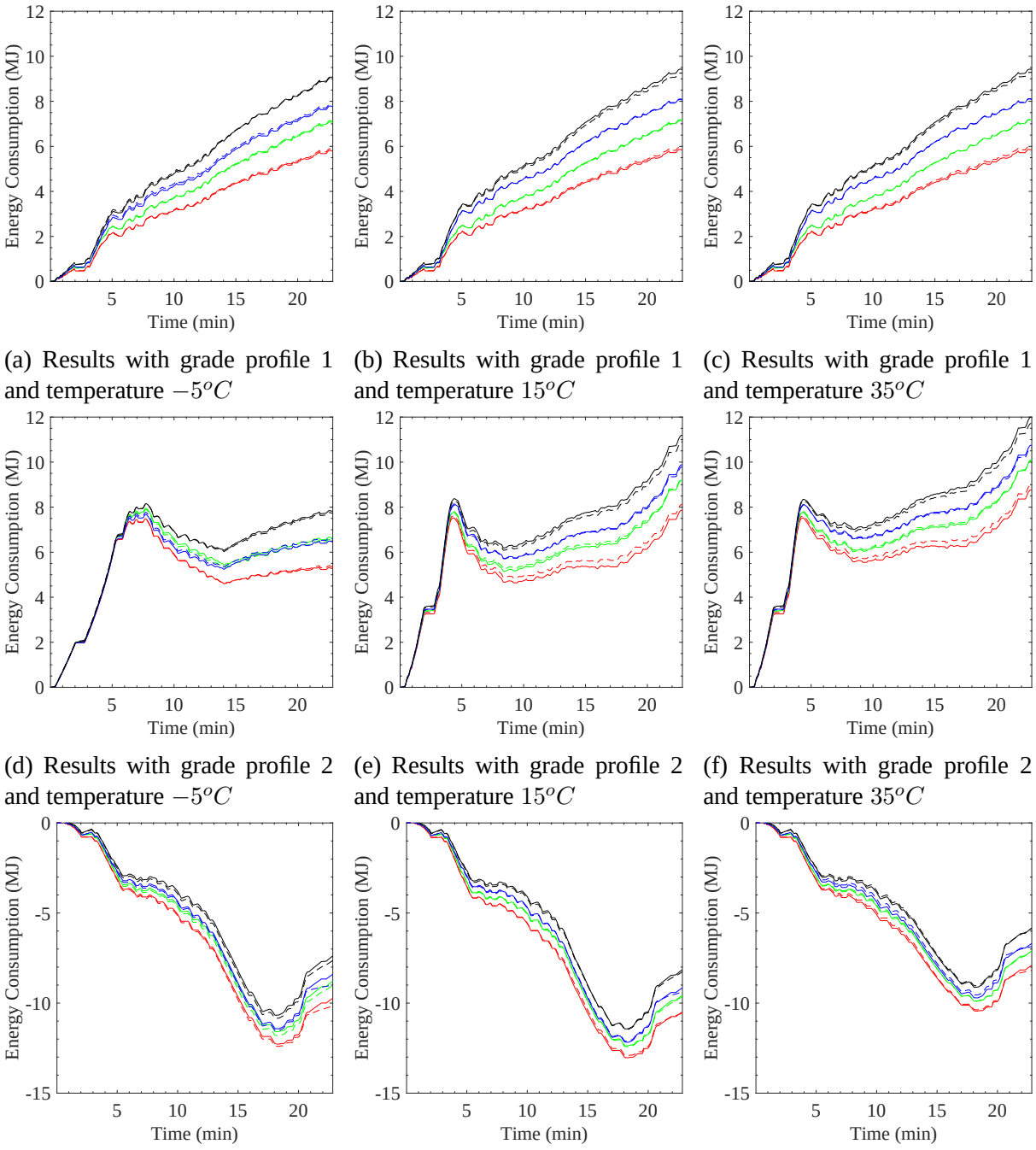
(g) Results with grade profile 3 and temperature -5^oC

(h) Results with grade profile 3 and temperature 15^oC

(i) Results with grade profile 3 and temperature 35^oC

Figure 4.3: Energy consumption estimation using IEE model for UDDS drive cycle at initial SOC of 30% under different conditions. Legend: Actual (Solid Line) / Predicted (Dashed Line) energy consumption under: air speed profile 1 with auxiliary load profile 1 (●) / auxiliary load profile 2 (●), air speed profile 2 with auxiliary load profile 1 (●) / auxiliary load profile 2 (●).

tion for air speed profile 2 and auxiliary load profile 2 after completing the drive cycle is about 9 MJ, 11 MJ and -8 MJ in Figure 4.4b, 4.4e and 4.4h, respectively. Energy consumption for grade profile 2 is more as compared to grade profile 1 because grade profile 1 has zero mean



(a) Results with grade profile 1 and temperature -5^oC

(b) Results with grade profile 1 and temperature 15^oC

(c) Results with grade profile 1 and temperature 35^oC

(d) Results with grade profile 2 and temperature -5^oC

(e) Results with grade profile 2 and temperature 15^oC

(f) Results with grade profile 2 and temperature 35^oC

(g) Results with grade profile 3 and temperature -5^oC

(h) Results with grade profile 3 and temperature 15^oC

(i) Results with grade profile 3 and temperature 35^oC

Figure 4.4: Energy consumption estimation using IEE model for UDDS drive cycle at initial SOC of 70% under different conditions. Legend: Actual (Solid Line) / Predicted (Dashed Line) energy consumption under: air speed profile 1 with auxiliary load profile 1 (●) / auxiliary load profile 2 (●), air speed profile 2 with auxiliary load profile 1 (●) / auxiliary load profile 2 (●).

and zero standard deviation i.e. no elevation/de-elevation whereas grade profile 2 has +ve mean i.e. mostly elevation. Similarly, energy consumption for grade profile 3 is -ve because mean grade for grade profile 3 is -ve i.e. de-elevation, so the energy will be generated which will be

used to charge the battery, hence -ve energy consumption.

Similar to road grade, the effect of temperature can be observed by considering one of the rows of the Figures 4.3 and 4.4, where only environment temperature is different for each sub-figure in a row and all other parameters are the same. For instance, consider the second row of Figure 4.3, the difference in energy consumption due to change in temperature is clearly visible. There is a peak at approximately 5 min in Figures 4.3e and 4.3f whereas no such peak exist in Figure 4.3d. The main reason for this is that the battery's performance degrades with decrease in temperature as battery's capacity decreases at low temperature due to increase in internal resistance of battery. Due to this, the battery can not provide sufficient power to the vehicle to reach the desired speed or acceleration and hence the driver is forced to run the vehicle at a lower speed. The peak in the figures is due to high speed/acceleration demand which was successfully fulfilled at temperature of $15°C$ and $35°C$ but at environmental temperature of $-5°C$ the battery was not able to provide sufficient power. This behaviour is normally seen when the battery's SOC is low as in this case where initial SOC is 30% but when the initial SOC is 70%, as in the second row of Figure 4.4, the energy consumption pattern at low temperature is similar to the energy consumption pattern at high temperature.

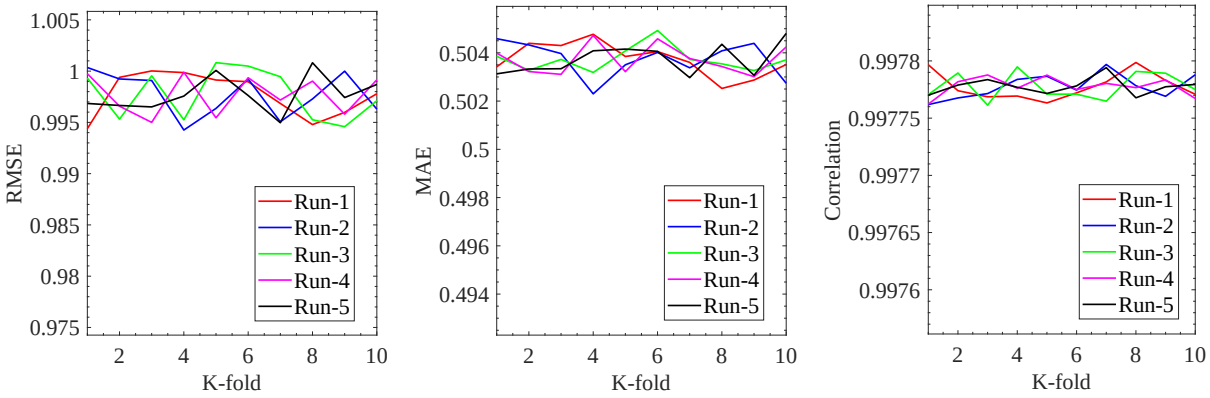
From the above discussion, it can be concluded that all of these parameters namely, vehicle speed, vehicle acceleration, road elevation, wind speed, auxiliary loads, environmental temperature and initial battery's SOC, are influencing the the energy consumption of EV significantly. Also from Figures 4.3 and 4.4, it can be observed that the IEE model successfully represents the non-linear influence of these parameters on energy consumption of EV with small deviation and can be used to estimate the energy consumption of EV in real-time.

4.3.1 Cross Validation of IEE Model

Cross validation of the IEE model has been performed to validate the generalizability and robustness of the methodology. There are several cross validation techniques and repeated k-fold cross validation is one of them, which is often used by researchers to validate their approach. In the current work, repeated 10-fold cross validation has been performed. The performance of the IEE model has been measured using three metrics namely, Root Mean Square Error (RMSE), Mean Absolute Error (MAE) and Correlation (Corr), defined in Section 3.3.4 using equations (3.9), (3.10) and (3.11), respectively.

In order to perform the 10-fold cross validation once for the IEE model, dataset $DS - I$ was divided into equally sized 10 small datasets. 70% of these equally sized 10 datasets were selected as the training set and remaining 30% were used as the validation set. Then the CNN

model of PCE Module and bagged decision tree of the Fine Tuner Module were trained using the training set and validated using the validation set. The process of selecting the training-validation set and training and validating the IEE model using the training-validation set was repeated 10 times (the folds). The datasets for training-validation set were selected, such that each of the 10 equally sized datasets is part of the validation set at least once. The process of 10-fold cross validation was repeated 5 times (the runs). The performance of the IEE model during the repeated 10-fold cross validation is shown in Figure 4.5. From the figure, it can be observed that the results from different runs overlap and also the results do not vary much. For instance, mean standard deviation in RMSE and MAE for different runs is 2.1×10^{-3} and 6.4×10^{-4}, respectively. So, it can be concluded that the IEE model performed consistently well during each run of repeated 10-fold cross validation with low variance.



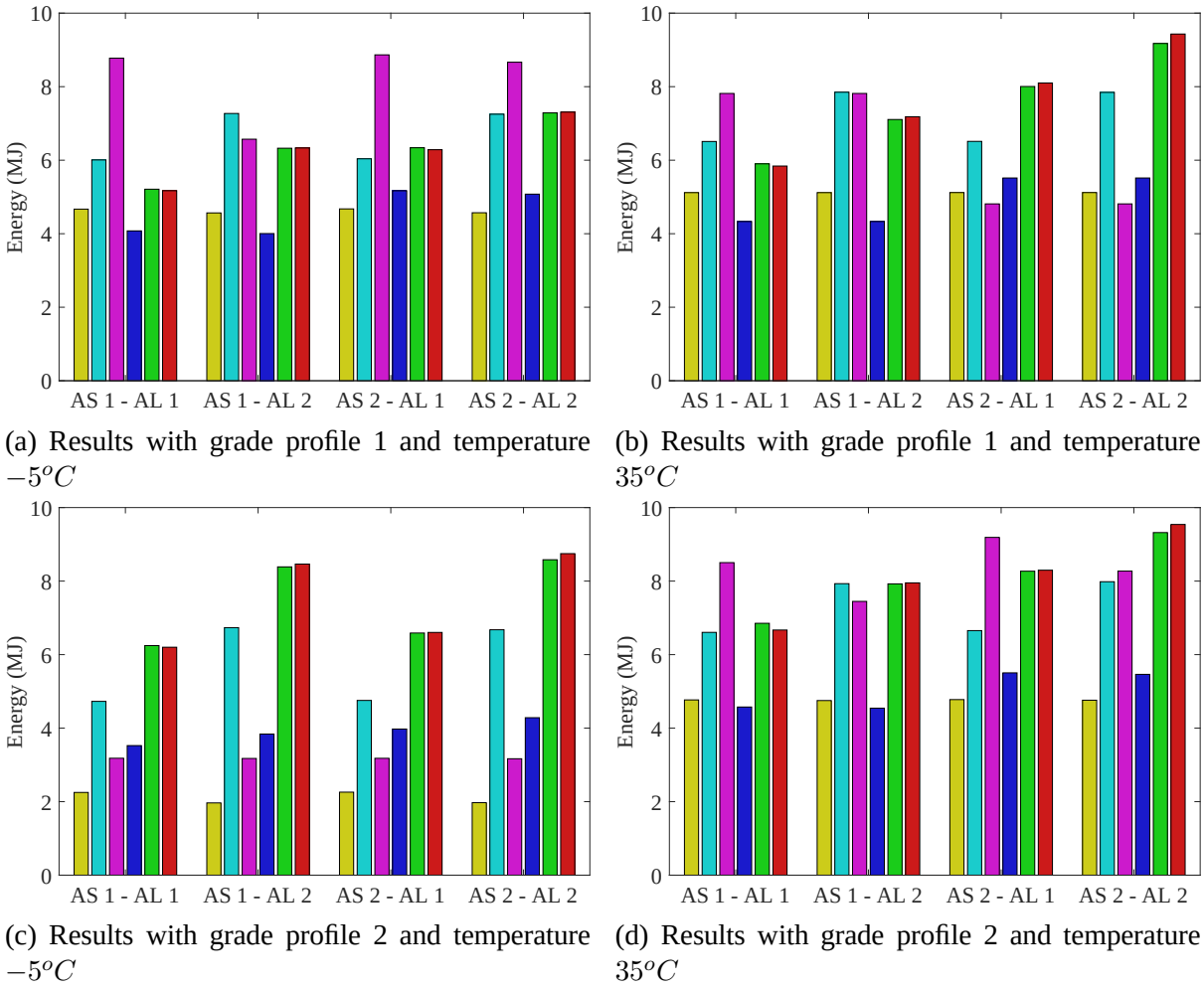
(a) RMSE (b) MAE (c) Correlation

Figure 4.5: Results of repeated 10-fold cross validation of IEE model using different error metrics for power consumption

4.3.2 Comparison of IEE Model with Other Existing Techniques

The results of three existing approaches, presented in [44, 66, 75], and the BEE model are compared with the IEE model to benchmark the results. The implementation of all these existing techniques have been explained in detail in Section 3.4 of previous chapter.

A comparison, for estimated energy consumption for UDDS drive cycle under different conditions, of the IEE, BEE model and three existing techniques is presented in Figures 4.6 and 4.7. Figure 4.6 shows the comparison of the results when the initial SOC of battery was 30% whereas Figure 4.7 shows the results with the initial SOC of 70%. In both figures, there are two rows and two columns, where each row depicts the results at different grade profiles and each column show the results at different environment temperatures. Each subfigure shows the comparison of the results with four combinations of two different air speed profiles (AS 1 and AS 2)



(a) Results with grade profile 1 and temperature -5^oC

(b) Results with grade profile 1 and temperature 35^oC

(c) Results with grade profile 2 and temperature -5^oC

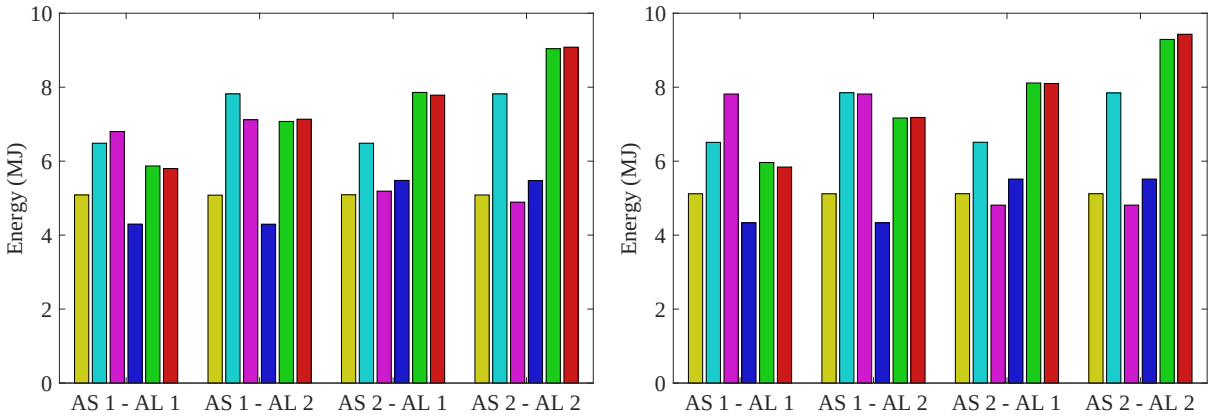
(d) Results with grade profile 2 and temperature 35^oC

Figure 4.6: Energy consumption prediction comparison of the IEE model with other existing techniques for UDDS drive cycle at initial SOC of 30% under different conditions of two road grade profile, two environmental temperatures, two air speed profiles (AS 1 and AS 2) and two auxiliary loads profiles (AL 1 and AL 2). Legend: Energy consumption: (■) Galvin [66], (■) Yang et al. [44], (■) Diaz Alvarez et al. [75], (■) BEE Model*, (■) IEE model†, (■) Actual.

* The model developed in Chapter 3

† The model developed in this chapter

and two different auxiliary load profiles (AL 1 and AL 2). The two different grade profiles, air speed profiles and auxiliary load profiles used to compare the results shown in Figures 4.6 and 4.7 are given in Table 4.1. From the figures, it can be observed that the IEE model gave consistently better energy consumption estimate compared to the existing techniques under all the different conditions considered. As the technique proposed by Galvin [66], considered speed and acceleration only to estimate the power/energy consumption it is justifiable for the results to deviate from the actual when other factors come into play. Similarly, Yang et al. [44] in their proposed technique does not consider the effect of environment temperature and initial SOC on energy consumption and hence the results are not as accurate but their technique performed much better than Galvin's technique. The neural network, proposed by Diaz Alvarez



(a) Results with grade profile 1 and temperature -5^oC

(b) Results with grade profile 1 and temperature 35^oC

(c) Results with grade profile 2 and temperature -5^oC

(d) Results with grade profile 2 and temperature 35^oC

Figure 4.7: Energy consumption prediction comparison of the IEE model with other existing techniques for UDDS drive cycle at initial SOC of 70% under different conditions of two road grade profile, two environmental temperatures, two air speed profiles (AS 1 and AS 2) and two auxiliary loads profiles (AL 1 and AL 2). Legend: Energy consumption: (■) Galvin [66], (■) Yang et al. [44], (■) Diaz Alvarez et al. [75], (■) BEE Model*, (■) IEE model†, (■) Actual.

* The model developed in Chapter 3

† The model developed in this chapter

et al. [75], performed badly for two main reasons, first it has no hidden layer and therefore was not able to learn the non-linearity among the influencing factors and second only three parameters namely, speed, acceleration and jerk were taken as input and the influence of other factors was not considered. The BEE model, gave consistently lower estimates than the actual energy consumption. The main reason for this is that it does not take into account the effect of varying auxiliary load, air speed, environment temperature and initial SOC of battery when estimating EVs energy consumption.

To generalize the comparison of the results, Table 4.2 shows the results of different performance metrics for the IEE and BEE models along with different existing techniques. The

different performance metrics are Mean Absolute Energy Deviation (MAE_{dev}), Root Mean Square Error ($RMSE$), Mean Absolute Error (MAE), Correlation ($Corr$) and Mean Prediction Time per Drive Cycle ($MPTDC$). MAE_{dev} can be obtained using the following equation:

$$MAE_{dev} = \frac{\sum_{i=1}^{n} \mid E_{act}^{i} - E_{est}^{i} \mid}{n} \tag{4.3}$$

where E_{act}^{i} and E_{est}^{i} represent the actual and estimated energy consumption for i^{th} drive cycle and n represents the total number of drive cycles under consideration. $RMSE$, MAE and $Corr$ are the same as defined in Section 3.3.4. $MPTDC$ is the average time an approach takes to predict the power consumption by the EV for a given drive cycle. It does not include the training time. It can be observed from the Table 4.2 that the IEE model gave very reliable results than the existing techniques. The IEE model has lowest MAE_{dev}, $RMSE$, MAE of 0.14, 0.97 and 0.50, respectively and highest $Corr$ of 0.997 for dataset $DS-I_{val}$ in comparison to the other existing techniques. Similar behaviour can be observed for dataset $DS-II$. As dataset $DS-II$ contains data recorded for road grade of 0% and no external wind, due to this all the approaches performed better on $DS-II$ compared to $DS-I_{val}$. Also, the deviation in prediction results by IEE model is very less compared to other techniques. For instance, the IEE model has least standard deviation of ± 0.069 MJ in absolute energy deviation for dataset $DS-II$ whereas Yang et al. [44], Galvin [66], Diaz Alvarez et al. [75] and BEE Model has standard deviation of ± 0.760 MJ, ± 1.320 MJ, ± 1.647 MJ and ± 1.314 MJ, respectively. The values of MAE, $RMSE$ and $Corr$ are not available for [75], as the NN proposed provides total energy consumed as output for the whole trip instead of providing instantaneous power consumption. Due to this, using NN model it is difficult to provide instructions in real-time to the driver based on the current power consumption. Also, there are a number of influencing parameters (including initial SOC of battery, wind speed and environmental temperature etc.) that need to be considered along with the parameters considered in [44, 66, 75]. Due to this, the IEE model gave reliable estimates even when these additional parameters come into play. The IEE model provides very good results but it takes more time compared to the existing

Table 4.2: Comparison of IEE and BEE models with existing techniques using different performance metrics

Approach	MAE_{dev}		$RMSE$		MAE		$Corr(R)$		$MPTDC$
	$DS-I_{val}$	$DS-II$	$DS-I_{val}$	$DS-II$	$DS-I_{val}$	$DS-II$	$DS-I_{val}$	$DS-II$	(in sec)
Yang et al. [44]	2.30	1.98	8.90	3.78	4.73	2.28	0.884	0.961	1.97×10^{-3}
Galvin [66]	6.77	2.75	13.79	4.04	8.64	2.64	0.377	0.970	$\mathbf{3.47 \times 10^{-4}}$
Diaz Alvarez et al. [75]	4.68	2.15	NA	NA	NA	NA	NA	NA	1.14×10^{-2}
BEE Model*	4.15	1.82	7.98	2.61	5.33	1.96	0.948	0.977	1.76
IEE Model†	**0.14**	**0.08**	**0.97**	**0.74**	**0.50**	**0.41**	**0.997**	**0.998**	2.10×10^{-1}

Values in bold represent the best ones and NA means not applicable

* The model developed in Chapter 3

† The model developed in this chapter

techniques, see the values of the fifth metric $MPTDC$, provided in the Table 4.2. The IEE model takes more time to predict results because it requires a lot of computation in the PCE and Fine Tuner modules. However, the objective of this work is to develop a technique which can provide more accurate results in comparison to other existing techniques. As the results presented in Table 4.2, are computed using system with 8 GB RAM and Intel i5 Processor, the real time performance can be achieved by converting the IEE model into TensorFlow Lite format and execute on Odroid boards with Movidius sticks or Google Coral boards. These are specially designed hardware components to run machine or deep learning models with high performance. For instance, Google Coral Dev board has Edge TPU coprocessor as an accelerator and can perform 4 Trillion Operations Per Second (TOPS) by consuming very less power of 0.5 watt for each TOPS.

4.3.3 Statistical Analysis

To further validate the conclusion that the IEE model is better than existing techniques, statistical analysis using two sample t-test also has been performed between the results of IEE model and other techniques i.e. IEE model with Galvin [66], IEE model with Yang et al. [44] and so on. For this, the 10 partitions used for validating the IEE model during repeated 10-cross validation were used and results were obtained for these partitions using other techniques. So, for each technique MAE_{dev}, $RMSE$ and other performance metrics were calculated for the results for each of the 10 partitions. Using these observations an analysis was performed by taking an assumption that with the statistical confidence level of $\alpha = 0.05$ the populations have equal variances. Population means difference as zero was taken as the null hypothesis. Under the above null hypothesis, two sample t-test was performed and Table 4.3 shows the results obtained. From the table, it can be seen that the values of the t-critical are less than the values of t-stat for each pair of techniques and also the p-values are less than the significance level $\alpha = 0.05$. This implies that the means of the populations differ significantly and hence null hypothesis is rejected. As the mean of the MAE_{dev} and $RMSE$ values is less for the IEE model than other techniques, it can be concluded that the IEE model provides better results and the difference in results is statistically significant.

4.4 Summary

In this chapter, a hybrid IEE model using CNN and BDT developed to provide the EV drivers with accurate energy consumption estimates in real-time has been presented. Hence, it can be used to guide the driver in real-time and decrease his/her range anxiety. The IEE model con-

Table 4.3: Results of state-of-the-art two sample t-test using MAE_{dev} and $RMSE$ of data partitions from 10-fold cross validation

	Galvin [66]	Yang et al. [44]	Diaz Alvarez et al. [75]	BEE Model*	IEE model†
	Results of t-test on MAE_{dev}				
Observations	10	10	10	10	10
Mean	6.772	2.306	4.679	4.155	0.139
Variance	29.660	4.104	16.258	4.710	0.012
Hypothetical mean difference	0	0	0	0	-
Pooled Variance	15.045	2.087	8.249	2.394	-
Degree of freedom	18	18	18	18	-
t–critical one tail	1.734	1.734	1.734	1.734	-
$P(T \leq t)$ one tail	0	0	0	0	-
t-stat	10.261	9.002	9.485	15.572	-
	Results of t-test on $RMSE$				
Observations	10	10	-	10	10
Mean	13.796	8.900	-	7.988	0.975
Variance	37.372	24.303	-	1.255	0.047
Hypothetical mean difference	0	0	-	0	-
Pooled Variance	20.788	13.528	-	0.724	-
Degree of freedom	18	18	-	18	-
t–critical one tail	1.734	1.734	-	1.734	-
$P(T \leq t)$ one tail	3.159×10^{-6}	6.902×10^{-5}	-	0	-
t-stat	6.287	4.817	-	18.433	-

* The model developed in Chapter 3

† The model developed in this chapter

siders the effect of seven factors namely, vehicle speed, acceleration, air speed, road elevation, auxiliary loads, environment temperature and the initial battery's SOC. As no internal vehicle parameters are required, the model can be easily generalized for any other electric vehicle. The accuracy of the IEE model has been validated by comparing the IEE model with other existing techniques. The comparison results show that IEE model can estimate the energy consumption with least mean absolute energy deviation of 0.08 ± 0.069 MJ and highest correlation of 0.998. So, from the comparison results, it can be concluded that the IEE model, unlike the previous existing techniques, can learn the non-linear relationships between different parameters and can provide improved estimates. The IEE model can be converted to TensorFlow Lite format and then can easily run on different microcontrollers or can be deployed to vehicular embedded system.

Chapter 5

Multistep Traffic Speed Prediction (MTSP) Model

Use of automobile vehicles is increasing at a very rapid rate which creates a number of social problems such as traffic congestion, overconsumption of energy, traffic accidents and high carbon emission [134]. To solve these problems Intelligent Transportation System (ITS) [135] have been considered as a promising solution. For example, ITS can provide travellers with real time information and can suggest optimal travelling routes based on future traffic predictions [136]. Other than this, based on the current traffic speed and future traffic change trend, ITS can help in optimizing the traffic signal timings which can ensure smooth traffic movement and hence, can help in reducing traffic congestion [137]. Also, for EVs traffic can play a vital role in the amount of energy consumption or regeneration. Therefore, it is necessary to predict the traffic speed in future so that the future energy consumption for a particular route can be calculated in advance and then using this information the EV driver can select the energy efficient route. In this chapter, a Multistep Traffic Speed Prediction (MTSP) system has been proposed to get the traffic status on a particular route or location in advance.

The chapter consists of four sections. Section 5.1, explains the architecture and working of the MTSP model. Section 5.2, discusses the datasets and hyperparameters used for training and the MTSP model. To benchmark the results, the MTSP model was compared with multiple other techniques. Section 5.3 provides the discussion and analysis of the comparison of the results with previous approaches. Finally, Section 5.4 summarizes the results and concludes the chapter. Note that, unless otherwise stated time is in minutes and traffic speed is in m/h.

5.1 Architecture of MTSP Model

The problem of the multistep traffic prediction (i.e. traffic prediction at multiple time steps into the future) has been addressed by developing a deep learning approach using the concept of latent space mapping. Figure 5.1, shows the framework of the MTSP model. Firstly, raw traffic speed data was collected from the loop detector sensors installed on city roads. These loop detectors can provide various traffic related parameters such as traffic speed, volume and flow

etc. In this work, historical traffic speed has been used to predict the future traffic speed. The data was pre-processed to make it suitable for use in training and testing the MTSP model. Certain data cleaning algorithms were employed for removing the various data anomalies, which occur due to physical limitations of loop detector sensors. After data cleaning, the neighboring sensors of all sensors were selected based on their different attributes (such as correlation and distance etc). Next, historical traffic data from all these neighboring sensors was formatted into a particular format to provide input to the MTSP model. The two auto-encoders, in the MTSP model, were trained separately: one for learning the pattern of the historical traffic speed and the other for future traffic speed. After the individual training of both of the auto-encoders, the pre-trained encoder of the first auto-encoder and the pre-trained decoder of the second auto-encoder was concatenated by adding a latent feature mapping layer in between them. Then, the concatenated model was optimized by training the latent feature mapping layer. Finally, the trained model was used to predict the future traffic speed. The following subsections discuss each part of the model in detail.

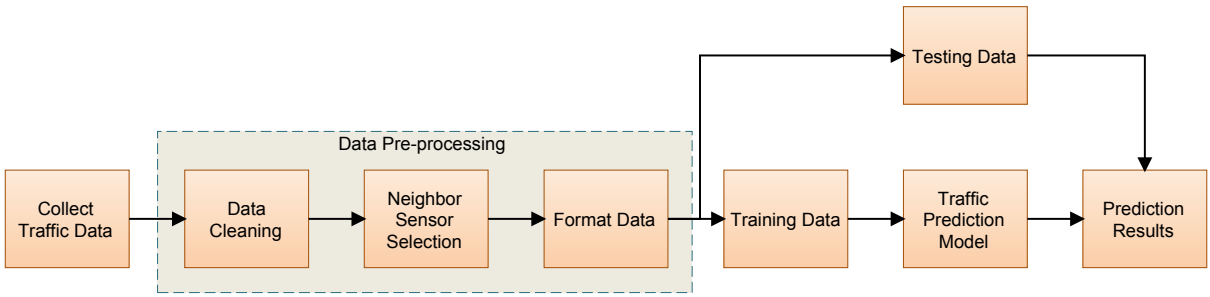


Figure 5.1: Framework of the MTSP model

5.1.1 Preprocessing of Traffic Data

Pre-processing of the collected traffic speed data was required before using the data for training or testing the MTSP model. It was achieved in three steps as discussed below:

5.1.1.1 Data Cleaning

Traffic speed data collected from various loop detector sensors contain a number of anomalies including missing values, due to failure of sensors, presence of outliers, due to noise, etc. Therefore the data was cleaned using data cleaning techniques before use. For instance, the loop detector sensors, for which the missing values were more than a particular threshold percentage, were removed from the dataset and for other loop detector sensors the missing values were replaced with values obtained by linearly interpolating the data. Similarly, the outliers (mainly point outliers i.e. single value which is too high or too low than its neighbors) in the dataset were replaced using the average value of the neighboring values.

5.1.1.2 Selection of Neighboring Sensors

Traffic on a particular road depends on the traffic status of neighboring roads. This is known as the spatial dependency of the traffic. The spatial dependency also changes with time, for instance, it might be possible at a particular point in time that more traffic is coming from one neighboring road and at other time from a different neighboring roads. Therefore, it is very important to cluster the traffic data according to the spatial and temporal dependency. For this, a neighbor sensor selection algorithm, as given below, has been used.

Algorithm 1: Algorithm for selecting the neighboring sensors

Input: $\mathbb{T}_{\mathbb{S}}$, $\mathbb{S}$, $\mathbb{D}$, p, t_0

Output: $\mathbb{S}^*$

Function `NeighborSensorSelection(`$\mathbb{T}_{\mathbb{S}}$, $\mathbb{S}$, $\mathbb{D}$, p, t_0`):`

initialize distance threshold δ

$S \longleftarrow \mathbb{S}[\mathbb{D}[p,:] < \delta]$ `// Get sensors with distance` $< \delta$ `// from` p

$D_S \longleftarrow \mathbb{D}[p, S]$ `// Get distance of sensors` S `// from` p

$T_S \longleftarrow \mathbb{T}_{\mathbb{S}}[t_0 - \Delta t : t_0, S]$ `// Get past traffic data of sensors` S `for` Δt `duration`

foreach $q \in S$ **do**

  $C_S[q] \longleftarrow corr(T_S[:,p], T_S[:,q])$ `// Find traffic correlation`

  $\triangle_S[q] \longleftarrow \left|\overline{T_S[:,p]} - \overline{T_S[:,q]}\right|$ `// Find absolute mean difference`

end

$\alpha_S \longleftarrow concatenate(C_S, D_S, \triangle_S)$ `// Concatenate the attributes`

$\xi \longleftarrow [+1, -1, -1]$ `// Attributes to be maximized(+) or minimized(-)`

$\lambda \longleftarrow [1, 1, 1]$ `// Weightage of the attributes`

$R_S \longleftarrow topsis(\alpha_S, \lambda, \xi)$ `// Rank the sensors using Topsis`

$\mathbb{S}^* \longleftarrow R_S[1:m]$ `// Select top` m `sensors`

return $\mathbb{S}^*$

end

The algorithm takes multiple input parameters, namely (i) list of all n sensors $\mathbb{S}$, (ii) distance matrix $\mathbb{D}$ of size $n \times n$, where each element $d_{i,j} \in \mathbb{D}$ represent the distance from sensor i to sensor j, (iii) traffic speed matrix of the sensors $\mathbb{T}_{\mathbb{S}}$ of size $\tau \times n$, where τ represents the number of time instances for which speed data was recorded and n is the number of sensors, (iv) sensor p for which neighboring sensors are to be identified and (v) time instant t_0. Firstly,

all the sensors which are less than δ distance apart from sensor p are selected and stored in S. Also, the distance of these sensors from p is stored in D_S. Then, traffic speed data of all the sensors S for time interval of $t_0 - \Delta t$ is obtained in T_S. Then, using the traffic speed data T_S, traffic similarity is calculated between p and each sensor $q \in S$. For traffic similarity, Pearson's correlation coefficient [138] and absolute mean traffic speed difference was used. So, for each sensor $q \in S$ three attributes were obtained i.e. distance between the sensors D_S, Pearson's correlation coefficient C_S and absolute mean traffic speed difference $\triangle_S$. The sensors in S were then ranked using a multi criteria decision making technique named Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) [139], based on these three attributes with equal weightage λ but with different decision making criteria ξ i.e. $+1$ for correlation because it should be maximum and -1 for distance and absolute mean difference because these two should be minimum. Then, top m sensors were selected as neighboring sensors $\mathbb{S}^*$ and returned by the algorithm.

5.1.1.3 Formatting the Traffic Data

After selecting the neighboring sensors for a particular sensor p at particular time instance t_0, past traffic data for all these neighboring sensors $\mathbb{S}^*$ was formatted in a specific format before using it for training and testing the MTSP model. The traffic data matrix (say, X of size $l \times m \times c$) was created from the past traffic speed data of all selected neighboring sensors. Here, l represent the number of time instances in the time interval Δt, m represent the number of selected nearby sensors and c is the number of channels. In this work, c was taken as 4 and each channel correspond to the traffic data of a different day, as shown in Figure 5.2. The first channel has data of the current day (say d), the second channel contains data for the past day i.e. $d - 1$ and third and fourth channels contain data for same day of the week as current day but of the past two weeks i.e. $d - 7$ and $d - 14$, respectively. This was done to capture the characteristics of the weekly traffic pattern. The time instances for which traffic data was used for each channel is also different i.e. the first channel contains data for time instances between $t_0 - \Delta t$ to t_0 whereas other channels contain data for time instances between $t_0 - \frac{\Delta t}{2}$ to $t_0 + \frac{\Delta t}{2}$. This helps the deep learning model in understanding how was the traffic speed pattern after time instant t_0 in past day and past two weeks. The traffic speed data matrix X was used as input for the MTSP model. The output of the traffic prediction model is future traffic speed vector $Y = [y_1, y_2, \cdots y_n]^T$ of sensor p. Here, y_i represents the traffic speed at sensor location p at time instance $t_0 + i$. The traffic data matrix X and Y were normalized into the range of [0,1] using the equation similar to Eq. (4.2). The normalized traffic speed data pair (X, Y) was then used for training and testing the MTSP model.

$$\begin{bmatrix} x^{d}_{t_0-\Delta t,1} & x^{d}_{t_0-\Delta t,2} & \cdots & x^{d}_{t_0-\Delta t,m} \\ x^{d}_{t_0-\Delta t+1,1} & x^{d}_{t_0-\Delta t+1,2} & \cdots & x^{d}_{t_0-\Delta t+1,m} \\ \vdots & \vdots & \ddots & \vdots \\ x^{d}_{t_0,1} & x^{d}_{t_0,2} & \cdots & x^{d}_{t_0,m} \end{bmatrix}$$

(a) $X(:,:,1)$

$$\begin{bmatrix} x^{d-1}_{t_0-\frac{\Delta t}{2},1} & x^{d-1}_{t_0-\frac{\Delta t}{2},2} & \cdots & x^{d-1}_{t_0-\frac{\Delta t}{2},m} \\ x^{d-1}_{t_0-\frac{\Delta t}{2}+1,1} & x^{d-1}_{t_0-\frac{\Delta t}{2}+1,2} & \cdots & x^{d-1}_{t_0-\frac{\Delta t}{2}+1,m} \\ \vdots & \vdots & \ddots & \vdots \\ x^{d-1}_{t_0+\frac{\Delta t}{2},1} & x^{d-1}_{t_0+\frac{\Delta t}{2},2} & \cdots & x^{d-1}_{t_0+\frac{\Delta t}{2},m} \end{bmatrix}$$

(b) $X(:,:,2)$

$$\begin{bmatrix} x^{d-7}_{t_0-\frac{\Delta t}{2},1} & x^{d-7}_{t_0-\frac{\Delta t}{2},2} & \cdots & x^{d-7}_{t_0-\frac{\Delta t}{2},m} \\ x^{d-7}_{t_0-\frac{\Delta t}{2}+1,1} & x^{d-7}_{t_0-\frac{\Delta t}{2}+1,2} & \cdots & x^{d-7}_{t_0-\frac{\Delta t}{2}+1,m} \\ \vdots & \vdots & \ddots & \vdots \\ x^{d-7}_{t_0+\frac{\Delta t}{2},1} & x^{d-7}_{t_0+\frac{\Delta t}{2},2} & \cdots & x^{d-7}_{t_0+\frac{\Delta t}{2},m} \end{bmatrix}$$

(c) $X(:,:,3)$

$$\begin{bmatrix} x^{d-14}_{t_0-\frac{\Delta t}{2},1} & x^{d-14}_{t_0-\frac{\Delta t}{2},2} & \cdots & x^{d-14}_{t_0-\frac{\Delta t}{2},m} \\ x^{d-14}_{t_0-\frac{\Delta t}{2}+1,1} & x^{d-14}_{t_0-\frac{\Delta t}{2}+1,2} & \cdots & x^{d-14}_{t_0-\frac{\Delta t}{2}+1,m} \\ \vdots & \vdots & \ddots & \vdots \\ x^{d-14}_{t_0+\frac{\Delta t}{2},1} & x^{d-14}_{t_0+\frac{\Delta t}{2},2} & \cdots & x^{d-14}_{t_0+\frac{\Delta t}{2},m} \end{bmatrix}$$

(d) $X(:,:,4)$

Figure 5.2: The channels of formatted traffic data matrix X

5.1.2 Traffic Prediction Model

A deep learning based model inspired from Deep Auto-Encoders (DAEs) has been developed for multistep traffic speed prediction. An auto-encoder has two main components, namely, an encoder and a decoder. The encoder is used to learn the internal representation of the input. This internal representation is also known as the latent space representation, which can also be used as a feature vector. The decoder does the reverse process and learns to convert the internal representation back to the input. Several researchers have used DAEs [140, 141] for robust feature extraction and then the extracted features have been used for classification or regression tasks. Vincent et al. [142] used auto-encoders to extract robust features by giving corrupted inputs. Similarly, Masci et al. [143] used the hierarchical features extracted using auto-encoders to initialize the convolutional neural network. Although, they have used auto-encoders for different problems, the effectiveness of auto-encoders for extracting the robust features has been demonstrated. Therefore, in this work pre-trained auto-encoders were used to predict the multistep traffic speed. Figure 5.3, shows the architecture of the traffic prediction model. Two different DAEs were used in the architecture: one DAE for historical traffic speed and the other DAE for future traffic speed. The encoder of the first DAE, E_X, converts the historical traffic speed X into the corresponding latent representation Z^X, i.e. $E_X : X \rightarrow Z^X$. On the otherhand, the decoder D_X recreates the historical traffic speed from its latent representation, i.e. $D_X : Z^X \rightarrow X$. Similarly, the encoder and decoder of other DAE converts future traffic speed Y to its corresponding latent representation Z^Y and then back to the future traffic speed.

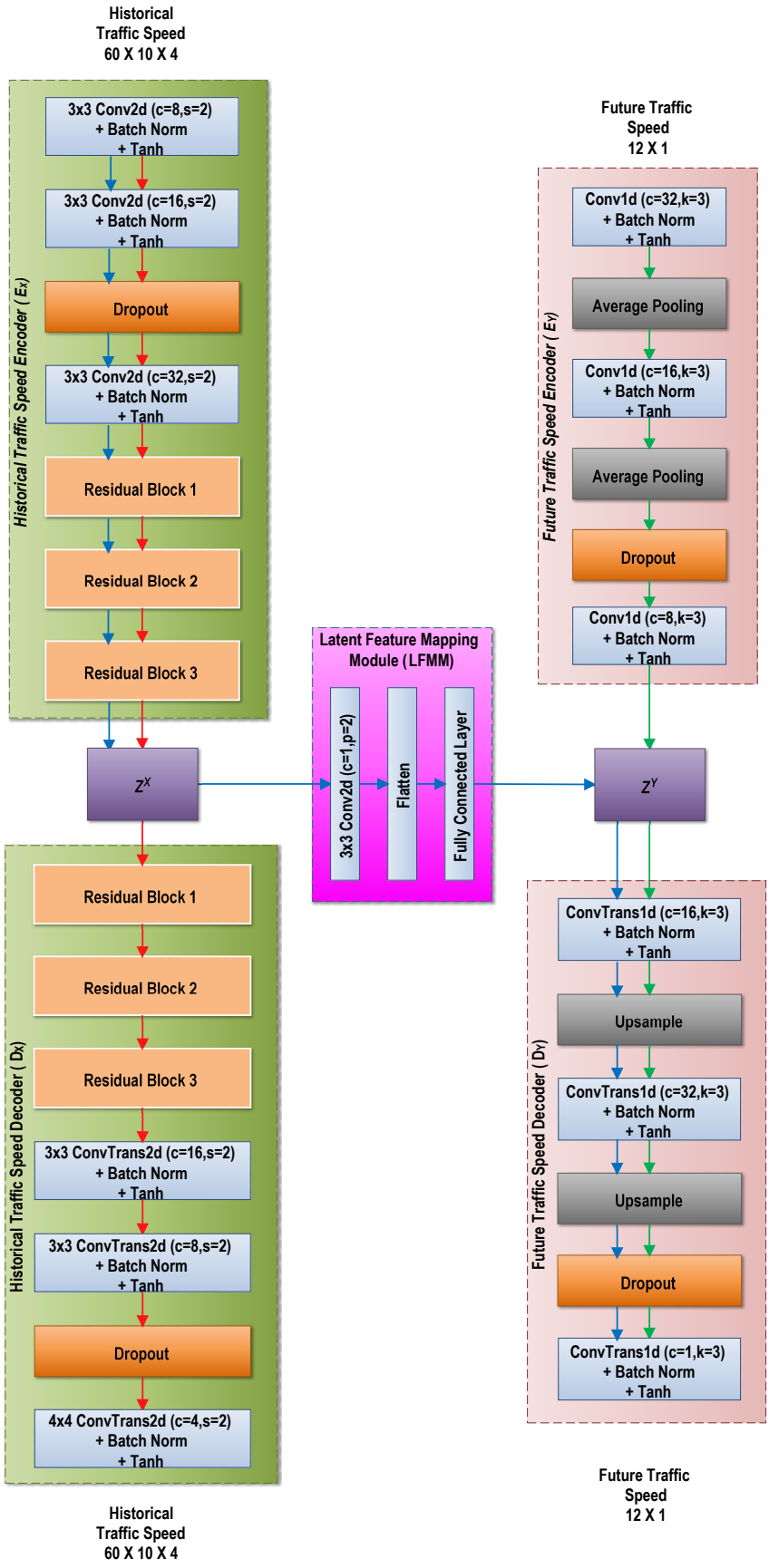


Figure 5.3: Architecture of Traffic Prediction Model

The extraction of the i^{th} latent feature map (l^i) for input I can be expressed as:

$$l^i = \sigma(I * W^i + b^i) \tag{5.1}$$

where σ, W^i, b^i and $*$ represent the activation function, weight, bias and convolution operation, respectively. Similarly, the reconstruction from latent representation is obtained using:

$$\hat{I} = \sigma\left(\int_{i \in L} l^i * \hat{W} + c\right) \tag{5.2}$$

where L, $\hat{W}$ and c respectively, represent the list of latent feature maps, transpose convolution operation and bias for input channel. The two DAEs try to extract the features of individual domains in terms of their corresponding latent representation. This is achieved by minimising the mean square error loss between input I and reconstructed output $\hat{I}$ of both DAEs which can be expressed as follows:

$$\mathcal{L}_{DAE} : f(I, \hat{I}) = ||I - \hat{I}||^2 \tag{5.3}$$

The idea was to use both the pre-trained DAEs, i.e. $DAE_X : E_X - D_X$ and $DAE_Y : E_Y - D_Y$, by cross connecting the encoder E_X with decoder D_Y for predicting the future traffic speed from historical traffic speed. The cross connection was achieved using Latent Feature Mapping Module $LFMM$, which maps the latent representation of historical traffic speed Z^X to the latent representation of future traffic speed Z^Y i.e. $FM : Z^X \rightarrow Z^Y$. As both the DAEs were trained separately, the two latent space representations i.e. Z^X and Z^Y do not lie in the same dimensional space. Therefore, the $LFMM$ fills the gap by properly mapping the two latent spaces. Since the encoder E_X was already trained to convert the historical traffic speed X into corresponding latent representation Z^X, it learns to effectively extract the critical features of X in the traffic prediction learning phase. Similarly, the decoder D_Y also learns to regenerate future speed Y from latent representation Z^Y while preserving the normal characteristics of Y due to the pre-training effect. In Figure 5.3, the flow of data for both the DAEs, i.e. DAE_X and DAE_Y is shown by red and green arrows, respectively. Similarly, the flow of data of cross-connected DAEs is represented using blue arrows. The following subsections discuss the architecture of both the DAEs and $LFMM$ in detail.

5.1.2.1 Deep Auto-Encoders (DAEs)

As discussed earlier, there are two DAEs (DAE_X and DAE_Y) in the architecture. The encoder of DAE_X has three convolutional layers followed by batch normalization and the Tanh activa-

tion layer. The three convolutional layers use convolutional kernels of size 3×3 with stride of 2 and padding of 1. Past research in deep learning has proved that the deeper networks can achieve better performance but simply stacking the layers makes the training process difficult due to the problem of vanishing gradient. So, to overcome this problem the concept of residual learning using residual blocks was introduced in [127]. Inspired by the success of residual learning, three residual blocks were used in the encoder E_X. Each residual block has two convolutional layers with kernel size of 3×3, stride of 1 and padding of 1 followed by batch normalization and non-linear activation ReLU. Each residual block has a connection which adds input of residual block to the batch normalized output of second convolutional layer of the block. To avoid overfitting of the model, a dropout layer was also used. The decoder D_X also has a similar structure but in the reverse order. To reverse the convolution effect, the transposed convolutional layers were used instead of convolutional layers.

The auto-encoder DAE_Y has simple architecture of convolutional auto-encoder. The encoder E_Y has three convolutional layers each followed by batch normalization and the tanh layer. Each convolutional layer has kernels of size 3, stride and padding of 1. Two average pooling layers also were used to maintain the average effect of features and reduce dimension. The decoder D_Y also has similar architecture with convolutional layers replaced with transposed convolutional layers and average pooling layers replaced with upsampling layers. Dropout layers were used both in encoder and decoder to avoid overfitting.

5.1.2.2 Latent Feature Mapping Module ($LFMM$)

The two DAEs were cross connected using $LFMM$, which maps the latent features Z^X of the historical traffic speed to the latent features Z^Y of future traffic speed. The $LFMM$ has three layers in sequence i.e. a convolution layer followed by a flatten and a fully connected layer. The convolution layer has convolutional kernels of size 3×3 with 1 stride and padding of 2 rows and 2 columns. As shown in Figure 5.3, the blue arrows show the flow of data for the cross connected network. The entire cross connected network was fine-tuned after pre-training the individual DAEs separately.

5.2 Experimental Setup for MTSP Model

This section describes the dataset used and initialization of the hyperparameters along with other details for training the MTSP model.

5.2.1 Description of Traffic Datasets

Traffic data from a very popular data source, available through a web-portal of California Department of Transportation Performance Measurement System (PeMS) [144] is used. The data source provides a large volume of traffic data from the loop detectors installed on the freeways of the state of California. Figure 5.4 shows the location of sensors on a map of Los Angeles, for which the traffic data is used for training and testing the MTSP model. There are 660 sensors on the mainline of freeways of Los Angeles, shown on the map with markers. Out of these sensors, 382 shown with green marker, were selected which cover the five freeways of Los Angeles, namely, I5, I10, I110, I405 and US101. The MTSP model was trained and tested to predict future traffic speed for these selected sensors. As explained in Section 5.1.1.2, the neighboring sensors for all these 382 sensors were obtained using Algorithm 1. The algorithm can select any sensor from all 660 sensors as the neighboring sensor depending upon the traffic pattern. For predicting traffic speed for the next one hour at a particular sensor the last five hours of traffic data was used. As traffic speed data at every 5-min interval is available from the data source and the top 10 sensors were selected using the neighbor selection algorithm, the input traffic speed matrix X has a dimension of $60 \times 10 \times 4$ and output traffic speed matrix Y has dimension of 12×1. Traffic data for the period of two months from 1st June, 2017 to 31st July, 2017 was used for the experiments. 70% of the dataset was used for training the MTSP model and the remaining 30% was used for testing. As traffic at night is very low and does not require forecasting, for the experiments in the current study traffic data from 7:00 am to 10:00 pm only was used.

5.2.2 Hyperparameters for Training MTSP Model

The MTSP model has two DAEs which were trained separately. Next, the two DAEs were cross connected using $LFMM$ and the fine tuning was carried out. For training the DAEs and the cross connected network, all the fully connected, convolutional and transposed convolutional layers were initialized with Xavier initialization [132]. It initializes the weight of a layer using random numbers from a uniform distribution with the limits of $\left[-\sqrt{\frac{6}{fan_{in}+fan_{out}}}, \sqrt{\frac{6}{fan_{in}+fan_{out}}}\right]$. Here, the variables fan_{in} and fan_{out} represent the count of input connections with the layer and count of output connections from the layer, respectively. The Adam optimization algorithm [133] was used for finding the optimized solution with initial learning rate of 0.001 and batch size of 128. The Adam algorithm at each step calculates exponential running average of the gradients and square of the gradients. To control the decay of these running averages the parameters β_1 and β_2 were initialized to 0.9 and 0.999, respectively. During training to avoid division by zero epsilon ϵ was set to a small value of 10^{-8}. The Mean Square Error

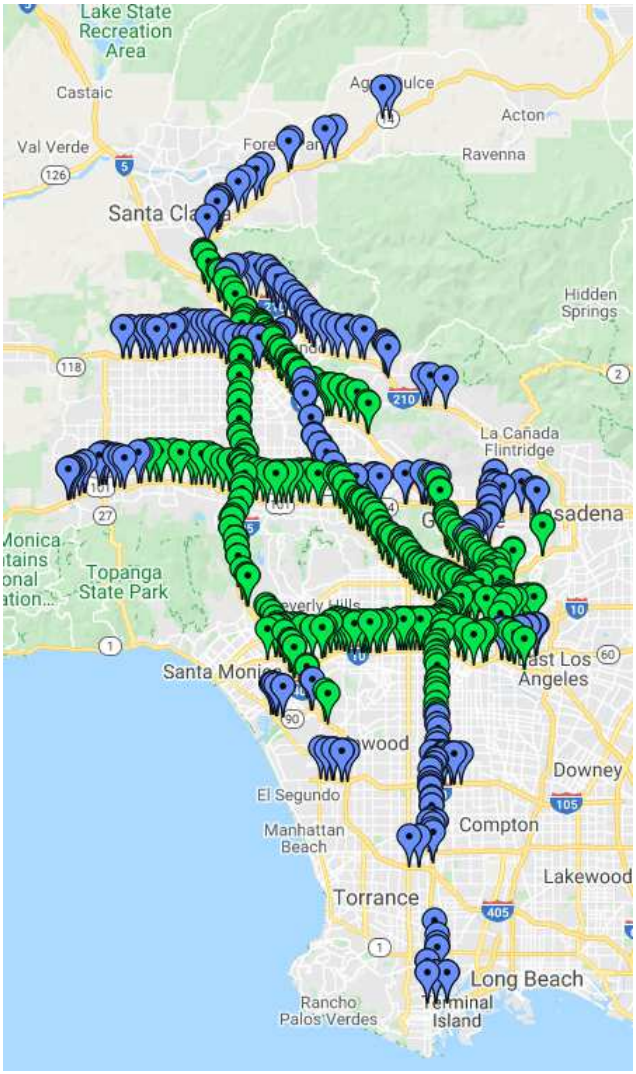


Figure 5.4: Loop detector sensors on the map of Los Angeles

(MSE) was used as the loss function during training. Also in the MTSP model there were some dropout layers which were used as regularizers to avoid overfitting. For all the dropout layers the dropout probability was set to 0.2, as the experiments show that increasing the drop rate further has negative effect on the model's performance.

5.3 Traffic Prediction Experimental Results

In this section, the traffic speed data collected from the loop detectors of Los Angeles is used to show the effectiveness of the MTSP model, by comparing the results with several state-of-the-art techniques which includes machine learning (ANN, kNN and XGBoost) and deep learning techniques (LSTM and CNN).

i) *ANN:* A three layered Artificial Neural Network (ANN) [145] was developed for traffic speed prediction. Each neuron in the ANN was activated using a sigmoid function. As ANN does not differentiate the input variables across time, it was not able to capture the temporal dependencies.

ii) *kNN:* A k-Nearest Neighbor (kNN) [146] approach finds the top k similar observations from the training set based on the Euclidean distance. Then, the future traffic speed prediction is calculated using the weighted sum of the corresponding future traffic speed of k selected observations. The hyper parameter k is selected using cross validation by varying the value of k from 5 to 20.

iii) *XGBoost:* XGBoost [147] is quite a popular machine learning technique which has already been applied to different tasks with outstanding performance. To implement XGBoost, all the input features were reshaped to a vector and then used as the input for training.

iv) *CNN:* Convolutional Neural Network (CNN) [104] has been applied to predict traffic speed by reshaping the traffic data into matrices. Although, CNNs are capable of learning complex patterns from the data, they can not take temporal dependency into account.

v) *LSTM:* Long Short Term Memory (LSTM) [148] is a very popular deep learning method which has been used widely for time series forecasting. LSTM is capable of taking temporal dependencies into account but can not capture the spatial information.

The most appropriate inputs were selected for the aforementioned techniques to ensure a fair comparison. The traffic data for the last 5 hours (i.e. past 60 time steps) of the target sensor was reshaped to form a vector and given as input to ANN, kNN, XGBoost and LSTM which provide a prediction of next 1 hour (i.e. 12 time steps) as output. All these models were trained with a number of such inputs for all the 382 sensors (mentioned in Section 5.2.1) taken at different time instances. The best hyperparameters for different models were chosen using cross validation, for instance, value of $k = 17$ for kNN gave best performance. For training the CNN, the same matrices were used, which were used for training the MTSP model. The CNN model contain three ReLU activated convolutional layers each followed by an average pooling layer and then a dense layer has been used to take the output. The input traffic data was normalized into the range of [0,1] before using for training the models.

The performance of the above said state-of-the-art techniques are compared with the MTSP model using three standard evaluation metrics, namely Mean Absolute Error (MAE), Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). The RMSE and MAE are already defined in Section 3.3.4 using Equations (3.9) and (3.10), respectively. The

MAPE can be calculated using the following equation:

$$MAPE = \left(\frac{1}{N} \sum_{i=1}^{i=N} \left| \frac{x_{act}^{i} - x_{pred}^{i}}{x_{act}^{i}} \right| \right) * 100 \tag{5.4}$$

where, N is the total number of samples, x_{act}^{i} and x_{pred}^{i} represent the actual and predicted value, respectively. The evaluation metric MAE and RMSE can measure the absolute deviation of the predicted values from the actual values whereas MAPE can measure the relative deviation. Table 5.1 shows the performance of the comparison of the results of the MTSP model and other benchmark techniques at five different prediction time horizons. From the table, it can be concluded that the MTSP model has the best performance with the least values at all of the prediction time horizons except for 5-min ahead prediction. Although the performance of the MTSP model is not the best for 5-min prediction but it is comparable to the performance of other techniques. Also, it can be observed from the table that as the prediction horizon increases the performance error for each technique also increases but consistently increase in error with prediction horizon is less for the MTSP model compared to other benchmark

Table 5.1: Traffic speed prediction performance comparison of MTSP model with other popular approaches

Technique	Metric	5-min	15-min	30-min	45-min	60-min
ANN	MAE	1.51	2.78	3.93	4.80	5.42
	RMSE	2.58	4.90	6.92	8.14	9.03
	MAPE	3.69	7.12	10.75	13.38	15.53
kNN	MAE	2.28	3.12	4.02	4.74	5.29
	RMSE	3.90	5.46	7.00	8.04	8.77
	MAPE	5.99	8.33	11.08	13.41	15.16
XGBoost	MAE	**1.27**	2.39	3.44	4.19	4.75
	RMSE	**2.13**	4.11	5.89	6.97	7.73
	MAPE	**3.09**	6.10	9.04	11.35	13.09
CNN	MAE	1.92	2.88	3.62	4.09	4.41
	RMSE	2.85	4.70	6.12	6.82	7.25
	MAPE	4.77	7.47	9.88	11.37	12.37
LSTM	MAE	1.52	2.62	3.43	3.96	4.55
	RMSE	2.48	4.38	5.85	6.73	7.55
	MAPE	3.71	6.56	8.87	10.43	12.11
MTSP*	MAE	1.64	**2.26**	**2.57**	**2.75**	**3.09**
	RMSE	2.58	**3.73**	**4.51**	**4.88**	**5.38**
	MAPE	4.43	**6.03**	**6.81**	**7.32**	**8.44**

Values in bold represent the best ones

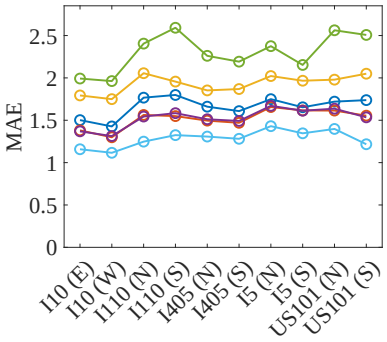
* The model developed in this Chapter

techniques. The main reason for this is the ability of the MTSP model to effectively capture spatial and temporal features of the traffic by selecting neighboring sensors at a particular instant based on traffic similarity and distance.

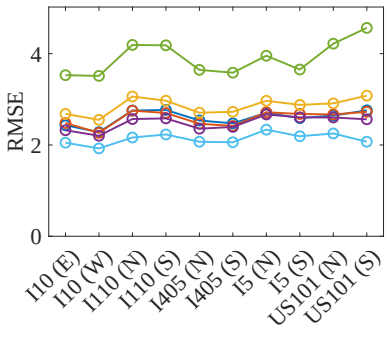
The traffic speed prediction performance of the MTSP model at multiple time horizons for each highway was also compared with other techniques and the results are shown in Figure 5.5. There are a total of 5 highways in the dataset i.e. I5, I10, I110, I405 and US101. The prediction performance for the traffic on both sides of the highway have compared separately. In Figure 5.5, the letter in round brackets along with highway name represent the traffic travel direction. The similar behaviour, as depicted in Table 5.1, can be observed in the figure. The MTSP model performed consistently better than other techniques on all highways with least error values. The consistency also can be observed by comparing the error with respect to time horizons. For instance, as shown in Figure 5.5i and 5.5l, the MAPE of MTSP model lies in the range of [5,10] approximately for both prediction time horizon of 30-min and 60-min whereas for same time horizons the MAPE of other techniques has increased from [7,15] to [10,21] approximately.

The difference in prediction performance of MTSP model and other technique is clearly visible in Figure 5.7. The figure shows the comparison of one day traffic prediction at three different time horizons (5-min, 30-min and 60-min) by MTSP and other benchmark techniques for two sensors 716960 and 717040 located on highway I5 and I10, respectively. The location of both the sensors on the map has been shown in Figure 5.6 for better understanding. In Figure 5.7, it can be seen that the MTSP model consistently provide traffic speed predictions which are very near to the actual traffic speed. The difference is more clearly visible in Figures 5.7e and 5.7f, where other techniques deviate from the actual traffic speed quite frequently while the MTSP model follows the pattern of actual traffic speed consistently. Also, it can be observed that the MTSP model is performing really well whenever there is sudden transition in traffic speed, for instance, at around 9:00 and 18:00 hours (peak hours) there is sudden change in traffic speed for both the sensors, as shown in Figures 5.7c, 5.7d, 5.7e and 5.7f, but the traffic prediction by MTSP model does not deviate much from the actual traffic speed as compared to other techniques.

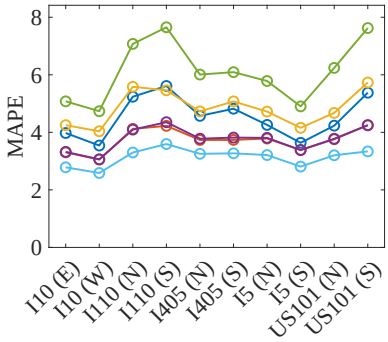
From the above discussion, it can be concluded that the MTSP model outperforms than the other state-of-the-art techniques for traffic speed prediction. The main reason for the improved performance is the capability of the MTSP model to understand the spatio-temporal dependencies for predicting the traffic speed. Also, the prediction performance degrades with increase prediction time horizons. The MTSP model provides better estimates even in peak hours where other techniques lag behind considerably.



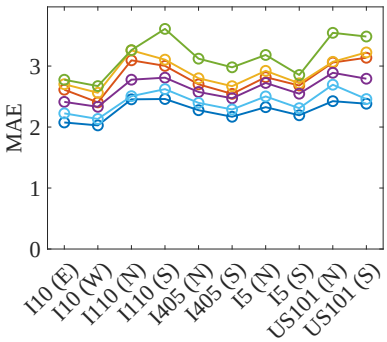
(a) MAE for prediction horizon of 5-min



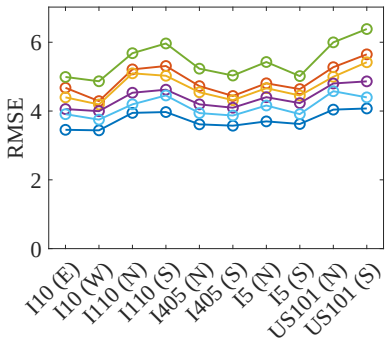
(b) RMSE for prediction horizon of 5-min



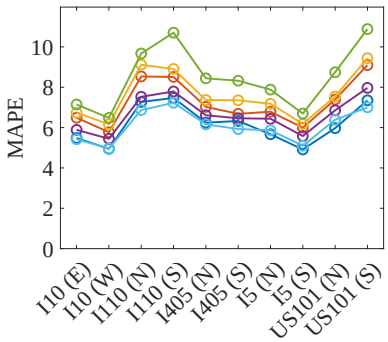
(c) MAPE for prediction horizon of 5-min



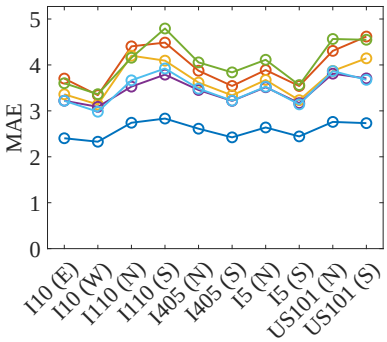
(d) MAE for prediction horizon of 15-min



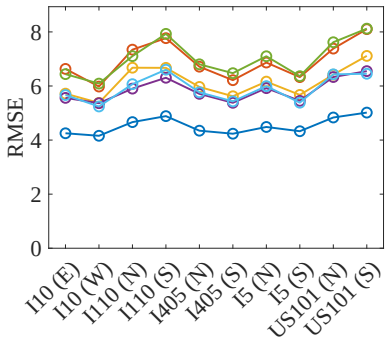
(e) RMSE for prediction horizon of 15-min



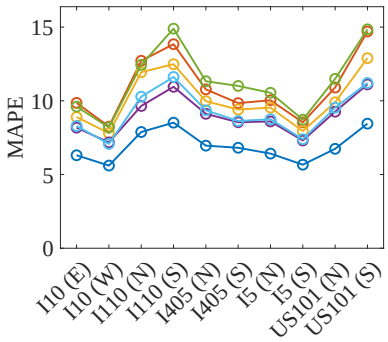
(f) MAPE for prediction horizon of 15-min



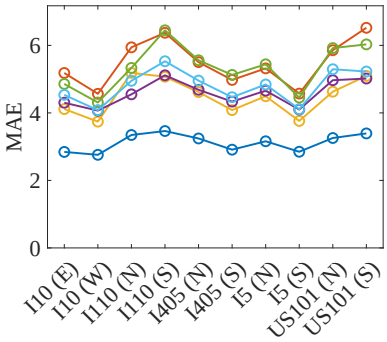
(g) MAE for prediction horizon of 30-min



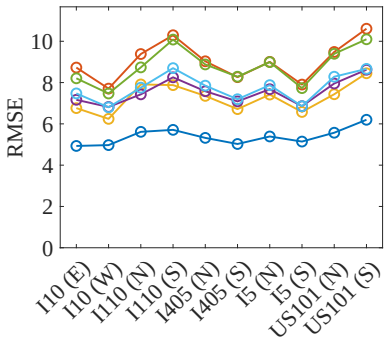
(h) RMSE for prediction horizon of 30-min



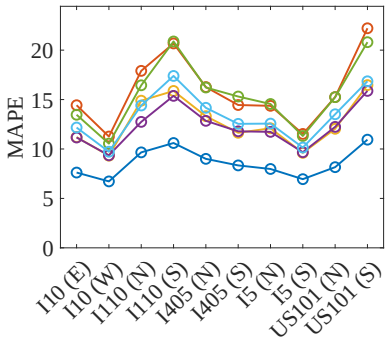
(i) MAPE for prediction horizon of 30-min



(j) MAE for prediction horizon of 60-min



(k) RMSE for prediction horizon of 60-min



(l) MAPE for prediction horizon of 60-min

Figure 5.5: Traffic speed prediction performance comparison of MTSP model with other techniques for each highway in the dataset. Legend: MTSP* (—), ANN (—), kNN (—), XGBoost (—), CNN (—), LSTM (—).

* The model developed in this chapter

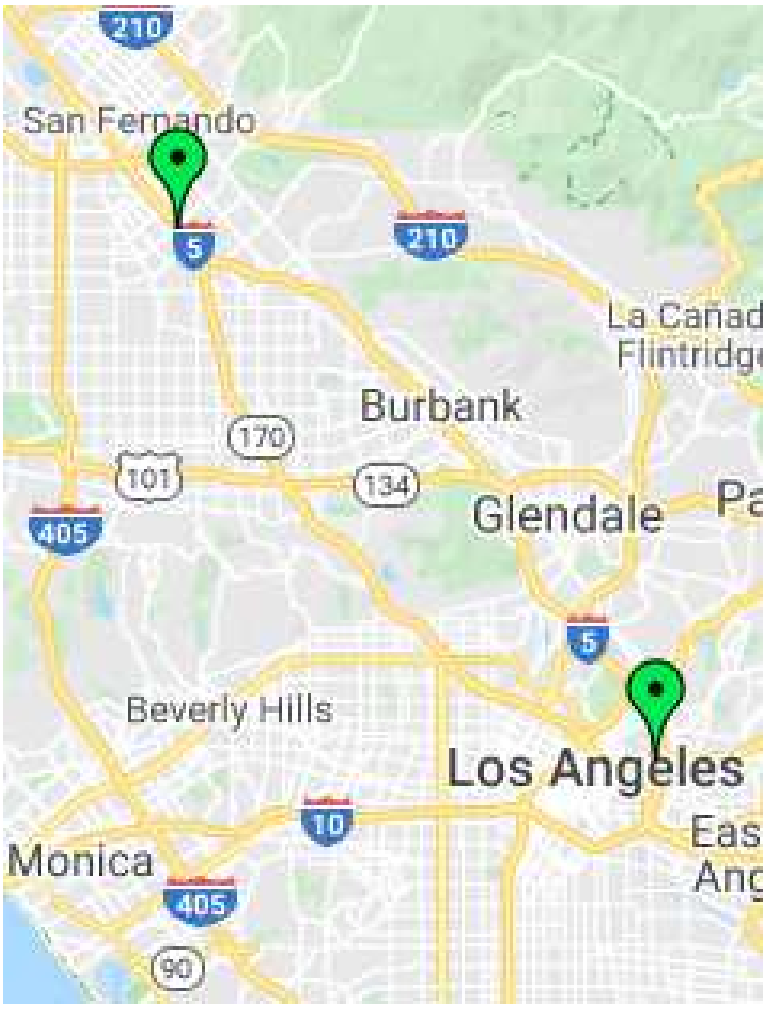
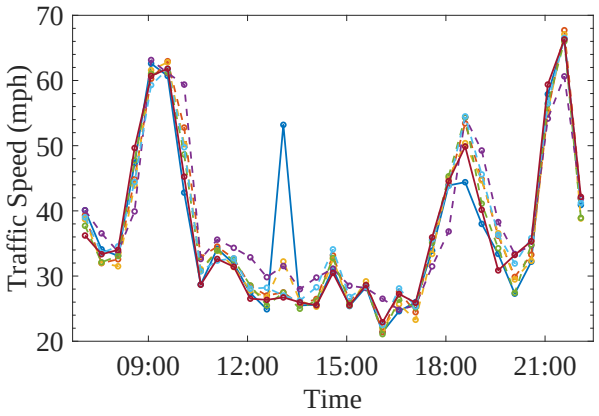


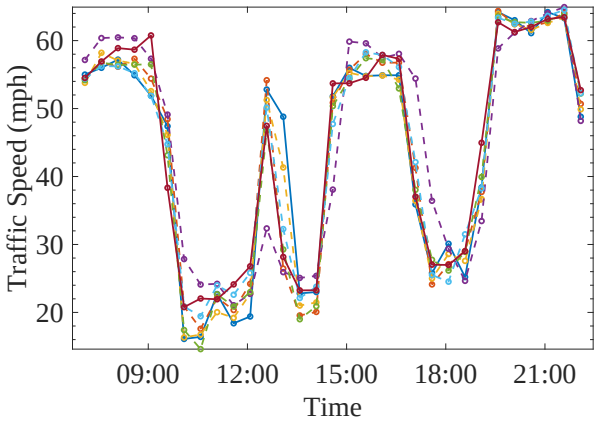
Figure 5.6: Location of sensors 716960 (top left) and 717040 (bottom right) on highway I5 and I10

5.4 Summary

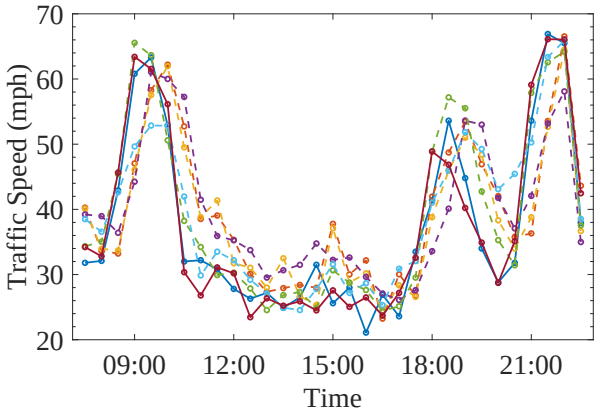
A deep learning based model has been developed which can provide multistep traffic speed prediction by considering the spatio-temporal traffic dependency. The MTSP model was trained with real-world traffic speed data, collected from a number of loop detector sensors located on different highways of Los Angeles. An algorithm has been developed to consider the spatio-temporal dependencies by selecting the nearby sensors based on traffic similarity and distance. The selection changes for each instant in time. The MTSP model contains two deep auto-encoders which were cross-connected using the concept of latent space mapping. The cross-connected auto-encoders were then trained using the historical traffic data from the selected nearby sensors. The effectiveness of the MTSP model was evaluated by comparing it with five



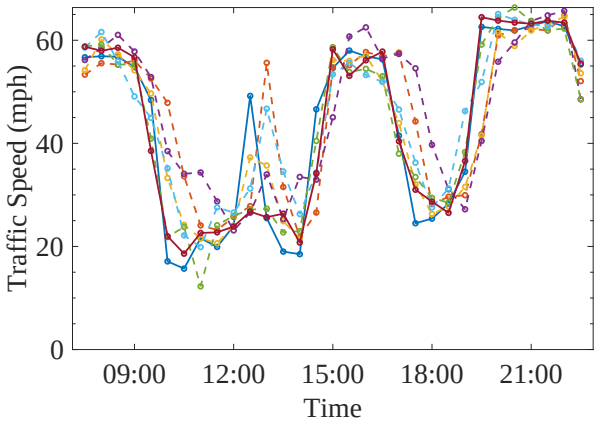
(a) 5-min traffic speed prediction for sensor 716960



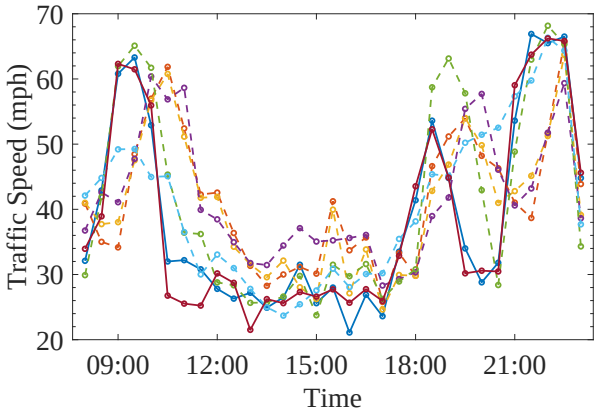
(b) 5-min traffic speed prediction for sensor 717040



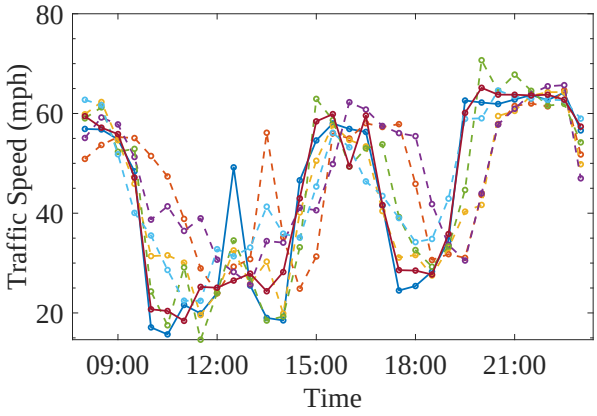
(c) 30-min traffic speed prediction for sensor 716960



(d) 30-min traffic speed prediction for sensor 717040



(e) 60-min traffic speed prediction for sensor 716960



(f) 60-min traffic speed prediction for sensor 717040

Figure 5.7: Prediction comparison of MTSP model with other techniques for traffic data of 25th July, 2017 from two sensors 716960 and 717040. Legend: Actual (—), MTSP* (—), ANN (- - -), XGBoost (- - -), kNN (- - -), LSTM (- - -), CNN (- - -).

* The model developed in this chapter

approaches (i.e. ANN, kNN, XGBoost, CNN, LSTM). A comparison of the results confirms that the MTSP provides better predictions with least error at different prediction time horizons. Although, the MTSP model has been trained and tested using the traffic data of highways of Los Angeles, it can easily be applied to any other city's traffic based on the availability of data.

Chapter 6

System Integration

The IEE model, discussed in Chapter 4, and MTSP model, discussed in Chapter 5, are integrated as a whole system. This chapter provides the details of the system architecture and discusses the results obtained from the integrated system. For this, the chapter has been divided into four sections. Section 6.1, explains the architecture of the integrated system. Section 6.2, discusses the testing procedure followed after testing the individual modules. Using the existing techniques multiple other integrated systems have been developed and compared. The comparison of the results are discussed in Section 6.3. Finally, Section 6.4 summarizes the findings and concludes the chapter.

6.1 Integrated System Architecture

An integrated system has been developed for improving the estimate of the energy consumption of electric vehicles. The system can be used 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. To achieve improved performance the system takes into consideration the effect of number of factors such as traffic, road elevation, wind speed, environmental temperature etc. Figure 6.1 shows the overall framework of the integrated system. It can be seen from the figure that the MTSP model, developed in chapter 5, is integrated with the IEE model, developed in chapter 4, by adding a module, named Speed Profile Generator, in between them. The integrated system will be referred to as the MTSP-IEE model, from now onwards. In brief, the data pre-processing module of MTSP model cleans, clusters and formats the raw traffic data and then the traffic prediction module predicts the future traffic speed. The predicted future traffic speed is then used by the speed profile generator module to predict the vehicle's future speed profile on a particular route and next the power consumption estimation module takes seven inputs of vehicle speed, acceleration and road elevation for a route etc. to predict the power consumption. The re-sampler module, up or down sample the different parameters to match the frequency of all the parameters before providing all the parameters as input to the fine tuner module. The fine tune module takes estimated power consumption along

with re-sampled seven input parameters of power consumption estimation module and fine tune the power consumption estimate. Finally, the state of charge calculator module calculates the remaining state of charge of the battery based on the estimated power consumption from fine tuner module. The algorithm for the generation of the vehicle speed profile from the predicted traffic speed by the speed profile generator module has been discussed in the following subsection.

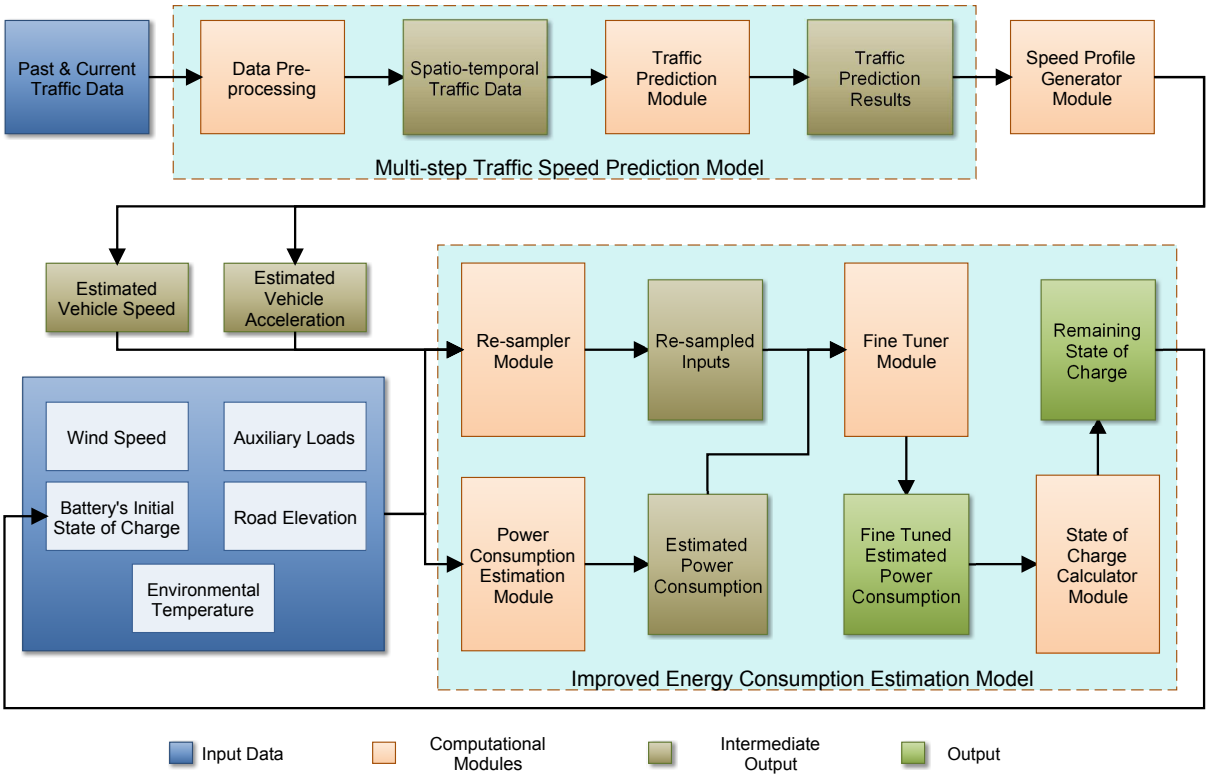


Figure 6.1: Architecture of integrated MTSP-IEE system

6.1.1 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) $\mathbb{S}_r$ which represents the list of k sensors on route r, (ii) $\mathbb{D}_r$ which represents the sequence of distances between the sensors i.e. the distance of second sensor from the first, then distance of the third from the 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 represents 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 the current speed v_{curr} at the current sensor and the next step's future speed at the 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. Finally, 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. The time series V_r after interpolation is used further by the IEE module for predicting the energy consumption for the particular route.

Algorithm 2: Algorithm for generating the future speed profile for a particular route

Input: $\mathbb{S}_r$, $\mathbb{D}_r$, Y_r, t_0
Output: V_r

Function `SpeedProfileGenerator(`$\mathbb{S}_r$, $\mathbb{D}_r$, Y_r, t_0`):`
  $T \longleftarrow 0$ // Initialize travel time T
  $V_r[p, t_0] \longleftarrow Y_r[p, t_0]$ // Get the speed of sensor p at time t_0
  $v_{curr} \longleftarrow Y_r[p, t_0]$ // Initialize current speed v_{curr}
  foreach $q \in \mathbb{S}_r$ **do**
    $j \longleftarrow \lfloor T/\Delta t \rfloor + 1$
    $d \longleftarrow \mathbb{D}_r[p, q]$ // Get the distance d to next sensor q from current sensor p
    $v_{avg} \longleftarrow \frac{v_{curr} + Y_r[q, t_0 + j \cdot \Delta t]}{2}$ // Calculate average speed of travel to next sensor q
    $t_d \longleftarrow \frac{d}{v_{avg}}$ // Calculate time to travel to q
    $v_{curr} \longleftarrow \frac{t_d \cdot Y_r[q, t_0 + j \cdot \Delta t] + (\Delta t - t_d) \cdot Y_r[q, t_0 + (j-1) \cdot \Delta t]}{\Delta t}$ // Update current speed v_{curr} with the future speed at time t_d at sensor q
    $T \longleftarrow T + t_d$ // Update total travel time T
    $V_r[q, T] \longleftarrow v_{curr}$ // Update V_r with current speed
    $p \longleftarrow q$ // Update current sensor p to q
  end
  return V_r
end

6.2 Testing the MTSP-IEE Integrated System

After training and testing the IEE model and MTSP model, they were integrated and tested as a complete MTSP-IEE system. For this, a directed node-edge graph G, as shown in Figure 6.2, was created where each node represents the sensor location on freeways of Los Angeles and directed edges represent the roads connecting these sensors. The direction of edges represents the direction of flow of traffic. The elevation at different sensor locations and the distances between them were obtained using free web APIs provided by the openroute service [149]. A number of routes were identified between different the origin and destination from the graph G.

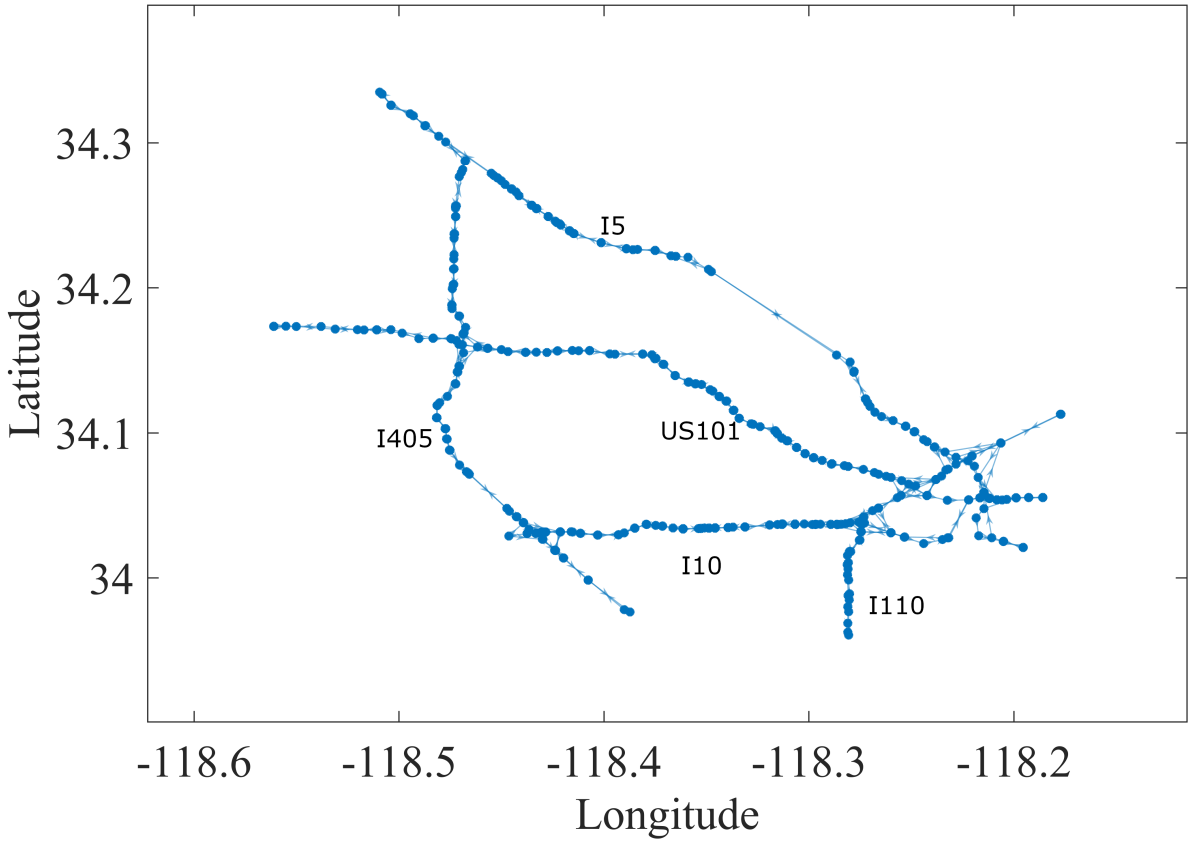


Figure 6.2: Node-edge graph G for loop detector sensors of Los Angeles

Environmental temperature and wind speed on different dates and time instances were obtained from free web APIs provided by Meteostat [150]. Finally, the system was tested to assess the accuracy of energy prediction along these routes at different time instances.

6.3 Discussion of Experimental Results of MTSP-IEE System

Multiple routes were identified with different origin and destination from the node-edge graph G, as explained in Section 6.2. Data from these routes at different time instances was used to test the performance of the MTSP-IEE system. The results of the MTSP-IEE system were benchmarked against multiple state-of-the-art techniques. For this three traffic prediction models, namely XGBoost, LSTM and CNN, were selected based on the comparison of their performance with MTSP model, as discussed in Section 5.3. Next, these selected models were integrated with two state-of-the-art techniques, presented in [44, 66], and the BEE model for energy consumption estimation and the results were obtained for all the routes selected from

graph G.

The integrated models of existing techniques, discussed above, were trained and tested using the data from different routes selected from graph G. The comparison of the results of MTSP-IEE system and these integrated models is presented in Table 6.1. The comparison was performed using three evaluation metrics i.e. RMSE, MAE and MAPE which are defined in Equations (3.9),(3.10) and (5.4), respectively. These metrics are calculated based on energy consumption estimation given by the particular integrated model and actual energy consumption given by FASTSim. In Table 6.1, the notation XGboost-Galvin represents an integrated model, in which XGBoost has been used for traffic speed prediction and then the model proposed by Galvin [66] has been used for energy prediction. Similar notation has been used for other integrated models. From the table, it can be concluded that the MTSP-IEE system provides results with the least RMSE, MAE and MAPE of 0.47, 0.32 and 1.60, respectively. Also, it can be observed that the integrated models, which uses the BEE model (model developed in Chapter 3) for energy consumption estimation, performed better than other integrated models. The main reason for this is that the model given by Galvin [66] does not consider the effect of other factors such as wind speed, road elevation and auxiliary loads etc. Similarly, the model proposed by Yang et al. [44] does not consider the effect of environmental temperature, battery's State of Charge (SOC) etc. Also, these models lack the ability to effectively represent the complex non-linear relationship between different influencing parameters. The BEE model 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 road elevation and backward force due to aerodynamic drag etc. Although, the effect of most

Table 6.1: Energy estimation performance comparison of MTSP-IEE system with other popular approaches

Technique	RMSE	MAE	MAPE
CNN-Galvin	9.25	7.68	29.57
CNN-Yang	6.67	5.32	19.68
CNN-BEE†	1.98	0.97	3.54
LSTM-Galvin	9.04	7.62	29.86
LSTM-Yang	6.51	5.35	20.57
LSTM-BEE†	0.99	0.65	2.55
XGBoost-Galvin	9.19	7.72	30.06
XGBoost-Yang	6.33	5.24	20.16
XGBoost-BEE†	1.53	0.85	3.09
MTSP-IEE*	**0.47**	**0.32**	**1.60**
Values in bold represent the best ones			

† The BEE model is developed in Chapter 3

* The integrated system developed in this chapter

of the factors have been considered, the effect of auxiliary loads, environmental temperature and battery's SOC was not considered in the CNN model of BEE model. It can be observed that the integrated model LSTM-BEE performed better than CNN-BEE and XGBoost-BEE. This is due to the better performance of LSTM than CNN and XGBoost, as shown in Table 5.1, for traffic speed prediction, in most of the cases i.e. traffic prediction for 15 min, 30 min and 45 min horizon. So, it can be concluded that the error accumulates when the different traffic prediction models were integrated with different energy consumption estimation models.

Similar observations can be drawn from Figure 6.3, which shows the performance comparison of MTSP-IEE system and other integrated systems for 100 randomly selected sample routes from graph G between different origin and destination pairs. The comparison has been performed by calculating the Absolute Percentage Energy Deviation ($APED$) for each route using the following equation:

$$APED = \left(\left| \frac{E_{act} - E_{est}}{E_{act}} \right| \right) * 100 \tag{6.1}$$

In above equation, E_{act} and E_{est} represent the actual and estimated energy consumption for a particular route. From the Figure 6.3, it can be observed that the MTSP-IEE system provide better estimates for all the routes with least $APED$. Also, for most of the routes the $APED$ is in the range of 0-5%, shown with black color, which is in accordance with the results presented

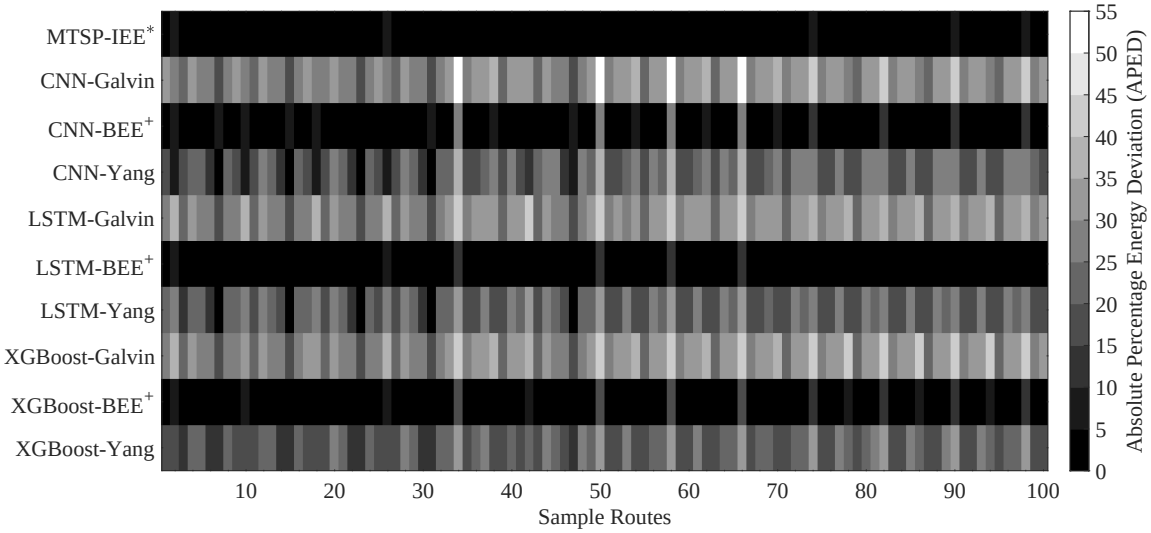


Figure 6.3: Performance comparison for 100 randomly selected sample routes (Dark color represent less error in energy consumption prediction)

* The integrated system developed in this chapter

+ The BEE model is developed in Chapter 3

in Table 6.1, as discussed above.

Out of these 100 randomly selected routes, two possible routes between location A and B are shown in Figure 6.4. One route, shown in blue color, is of approximately 7.7 miles and other route, part of which is shown in grey color, is of approximately 8.7 miles. Both the routes have a common portion at the start then deviate. Henceforth, the shorter route will be represented by $R-I$ and longer route will be represented using $R-II$. The energy prediction comparison of the MTSP-IEE system and other integrated models for route $R-I$ on the 13^{th} July, 2017 has been presented in Figure 6.5. From the figure, it can be observed that the MTSP-IEE system shows the same behaviour as the actual i.e. the deviation of MTSP-IEE system's estimates from the actual is much 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 MTSP-IEE system and the integrated models gave different traffic speed estimates and hence follow a different speed profile. Due to this, the estimated energy consumption varies a lot for the different approaches, but the MTSP-IEE system performed better compared to all the integrated models.

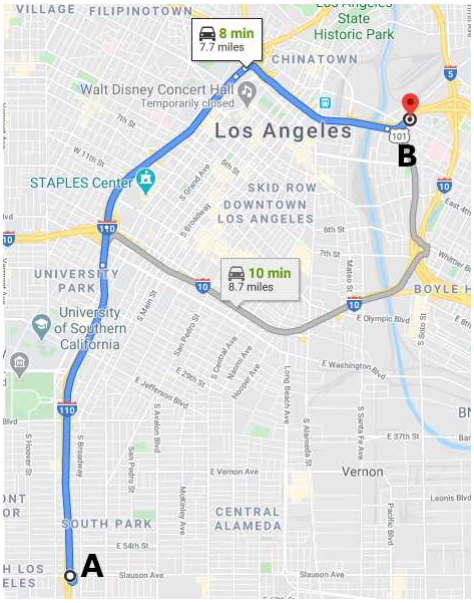
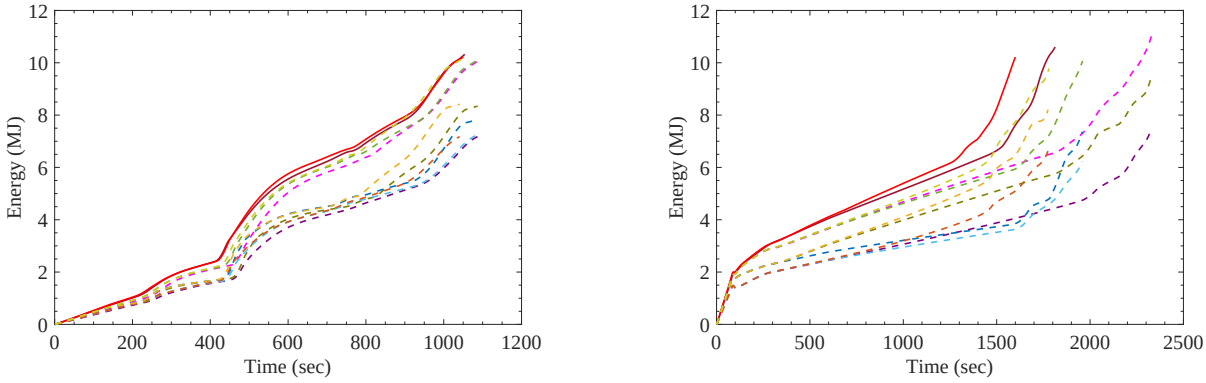


Figure 6.4: Two routes from location A and B

The comparison of energy estimation by MTSP-IEE system and different integrated models for routes $R-I$ and $R-II$ for two different days i.e. 13^{th} July, 2017 (Thursday) and 16^{th} July, 2017 (Sunday) has been presented in Figure 6.6. From the figure, it can be observed that the energy consumption is higher on the weekend in the morning of 16^{th} July, 2017 (Sunday) at 10 AM 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



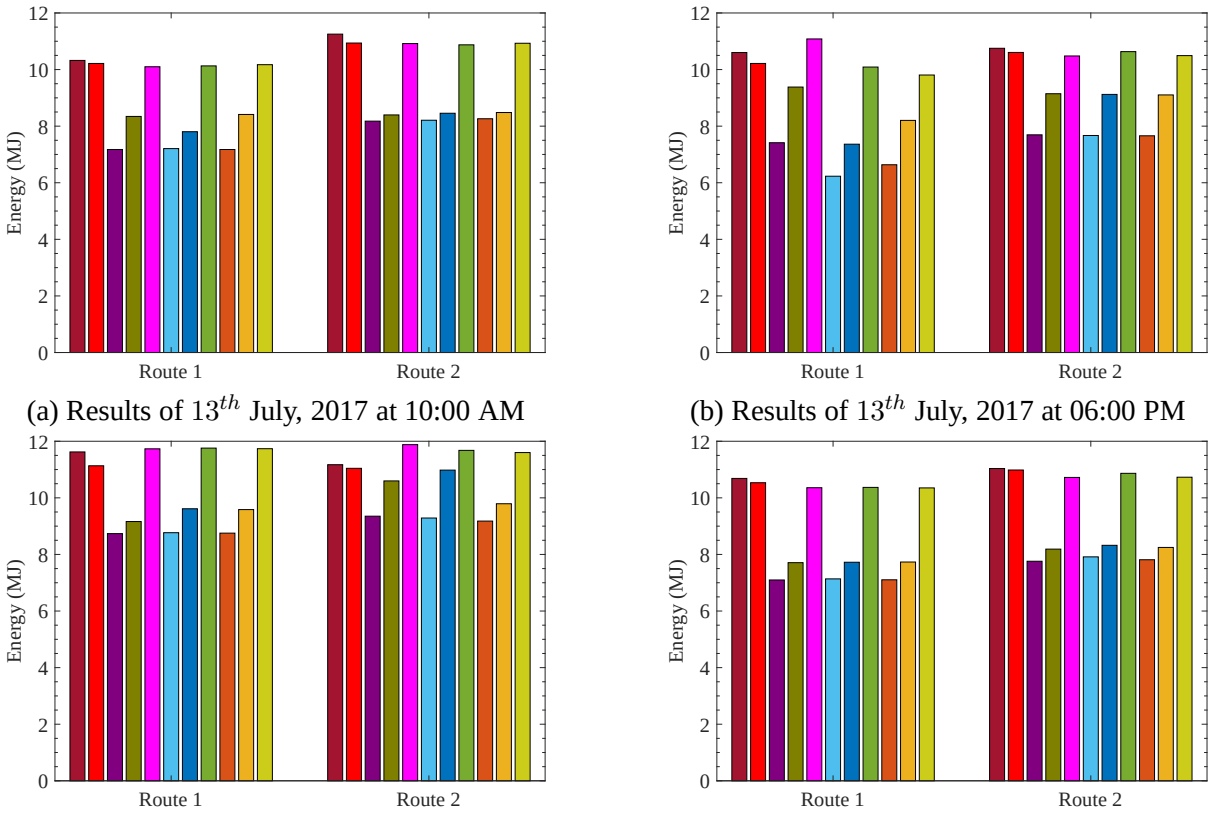
(a) Results at 10:00 AM

(b) Results at 06:00 PM

Figure 6.5: Performance comparison of MTSP-IEE system with other techniques on route $R-I$ for single day i.e. 13^{th} July, 2017 at two different time instants (10:00 AM and 06:00 PM). Legend: Actual (—), MTSP-IEE* (—), CNN-Galvin (- - -), CNN-Yang (- - -), CNN-BEE† (- - -), LSTM-Galvin (- - -), LSTM-Yang (- - -), LSTM-BEE† (- - -), XGBoost-Galvin (- - -), XGBoost-Yang (- - -), XGBoost-BEE† (- - -).

* The integrated system developed in this chapter

† The BEE model is developed in Chapter 3



(a) Results of 13^{th} July, 2017 at 10:00 AM

(b) Results of 13^{th} July, 2017 at 06:00 PM

(c) Results of 16^{th} July, 2017 at 10:00 AM

(d) Results of 16^{th} July, 2017 at 06:00 PM

Figure 6.6: Performance comparison of MTSP-IEE system with other techniques on two routes from location A to B for two different days (13^{th} and 16^{th} July, 2017) and two different time of day (10:00 AM and 06:00 PM). Legend: Actual (■), MTSP-IEE* (■), CNN-Galvin (■), CNN-Yang (■), CNN-BEE† (■), LSTM-Galvin (■), LSTM-Yang (■), LSTM-BEE† (■), XGBoost-Galvin (■), XGBoost-Yang (■), XGBoost-BEE† (■).

* The integrated system developed in this chapter

† The BEE model is developed in Chapter 3

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 the longer distance to be covered in route $R-II$, but in Figure 6.6c the energy consumption for route $R-II$ is predicted to be 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 low already. This is just one case but there can be other such cases where the driver can save more energy. So, in such cases, using the estimates provided by the MTSP-IEE system, the driver can select the route $R-II$ over $R-I$ for saving energy even though the vehicle has to cover more distance.

6.4 Summary

A integrated MTSP-IEE system has been developed which can provide the accurate and real-time energy consumption estimates to the EV drivers. The MTSP-IEE system can provide reliable real-time energy consumption estimates and can be used for guiding the EV drivers in real-time, thus can reduce driver's range anxiety. The MTSP-IEE system takes into account the influence of traffic, wind, road elevation, temperature, battery's SOC and auxiliary loads. The system can be generalized for other vehicles as no internal vehicle parameters are required from manufacturer to train and use the proposed system. The comparison of MTSP-IEE system with other benchmark techniques validates that the MTSP-IEE system performs better than other techniques with least mean absolute percentage error of 1.60%.

Chapter 7

Conclusion and Future Work

In this chapter, the work presented in the thesis has been concluded by highlighting its significance. For this, the chapter has been divided into two sections. Section 7.1 summarizes the work presented in this thesis by highlighting the main points of the individual modules and hence the complete integrated system. Section 7.2, provide some future directions to extend the work.

7.1 Conclusion

The work presented in this thesis aims towards developing a system that can provide guidance to the EV driver to minimize the energy consumption for reaching a particular destination based on current charge present in battery and different traffic, road and environmental conditions. The system is capable of providing real time energy consumption estimates for different routes to the destination and hence can be used by the drivers to select the best possible route based on predicted energy consumption and time. For this, a system has been developed, as discussed below, by integrating two main modules namely, the energy consumption estimation module and the traffic speed prediction module.

1. *Energy consumption estimation module*: Two deep learning based solutions were developed, capable of estimating the energy consumption for an EV based on different influencing factors to a known level of confidence. The first solution namely, the Basic Energy Estimation (BEE) model (developed in Chapter 3), uses a CNN for making the estimate based on three parameters namely, tractive effort, road elevation and vehicle's speed. The second solution namely, the Improved Energy Estimation (IEE) model (developed in Chapter 4), is the improved version of the first solution and uses a multi-channel CNN along with Bagged Decision Tree (BDT) to estimate the energy consumption by considering a number of other factors such as wind speed, environment temperature, battery's State of Charge (SOC) and auxiliary loads etc. The following points provide the main highlights of the developed energy consumption solutions:

(a) The proposed solutions can provide energy consumption estimates with known level of confidence in real-time. Hence, they can be used to guide the driver in real-time and decrease his/her range anxiety.

(b) The proposed solutions does not require any vehicle specific parameters, such as battery's internal resistance, the motor's efficiency curve, the battery's discharging/charging curve etc., for estimating the energy consumption. Due to this, the proposed approach can be easily generalized for any other electric vehicle.

(c) The accuracy of the proposed approaches has been validated by comparing them with other existing techniques. The comparison of the results of different approaches show that the IEE model can estimate the energy consumption with a least Mean Absolute Energy Deviation (MAE_{dev}) of 0.08 ± 0.069 MJ and highest correlation of 0.998.

(d) From the comparison of the results, it can be concluded that both of the solutions developed in this research, unlike the previous existing techniques, can learn the non-linear relationships between different parameters to improve the estimate of the energy consumption. The improvement can be seen based on the values of different error metrics, presented in Table 4.2, for instance, MAE_{dev} for test dataset has decreased from 1.98 MJ for Yang et al. [44] to 0.08 MJ for IEE model.

2. *Traffic speed prediction module*: A deep learning based approach namely, Mutistep Traffic Speed Prediction (MTSP) model, has been developed in Chapter 5 which can provide traffic speed prediction at multiple time-steps ahead by considering the spatio-temporal traffic dependency. The MTSP approach contains two deep auto-encoders which were cross-connected using the concept of latent space mapping. The following are the main highlights of the proposed approach:

   (a) The MTSP approach can provide traffic speed prediction multiple time-steps ahead based on the historical traffic data. The model is able to predict traffic speed for 5-min ahead horizon with MAE of 1.64 m/h and the error in prediction increases by approximately 6% averagely with each 5-min increase in prediction horizon.

   (b) The MTSP approach was trained with real-world traffic speed data collected from a number of loop detector sensors located on different highways of Los Angeles. The spatio-temporal dependencies were taken into account by selecting the nearby sensors based on traffic similarity and distance. The selection changes for each time instance.

(c) The effectiveness of the MTSP technique is validated by comparing it with 5 previous approaches (i.e. ANN, kNN, XGBoost, CNN, LSTM). The comparison of the results confirms that the MTSP technique provides improved prediction with the lowest of errors at all prediction time horizons, except 5 minutes.

(d) Although, the MTSP approach has been trained and tested using the traffic data of highways of Los Angeles, it can easily be applied to any other city's traffic based on the availability of data.

7.2 Limitations and Future Research

The present work focuses on developing a system to predict the energy consumption based on different influencing factors to provide real time guidance to the EV driver. For this, deep learning models have been developed using torch/pytorch and MATLAB environments. Future research could usefully convert these models into TensorFlow Lite format such that they can be used with Google Coral boards or Odroid boards with Movidius sticks etc. and hence, can be deployed in vehicle's embedded system to provide the optimal driving parameters (such as speed, the route to be taken etc.) to the driver in real time. Also, in future more factors like wind direction, passenger/luggage weight and the battery's state of health can be considered for estimating energy consumption of an EV. The proposed traffic prediction technique can be further extended to predict long-term traffic speed such as next day traffic speed prediction. Although, the proposed traffic prediction approach consider the effect of environment conditions or social events implicitly by neighboring sensor selection algorithm, the prediction accuracy can be improved by considering these factors explicitly. Also, along with pre-trip speed prediction if information about time for covering the distance is also made available to the driver before starting the trip then, the driver can take better decisions in selecting the alternative routes.

References

[1] OICA, 2005-2019 Sales Statistics, OICA, 2020. URL: http://www.oica.net/category/sales-statistics/. Accessed on: 24^{th} Sept, 2020

[2] J. Sousanis, World Vehicle Population Tops 1 Billion Units, WardsAuto, 2011. URL: https://www.wardsauto.com/news-analysis/world-vehicle-population-tops-1-billion-units. Accessed on: 24^{th} Sept, 2020

[3] D. Campbell-Lendrum, A. Prüss-Ustün, Climate change, air pollution and noncommunicable diseases, Bulletin of the World Health Organization 97 (2019) 160–161.

[4] M. Cepeda, J. Schoufour, R. Freak-Poli, C. M. Koolhaas, K. Dhana, W. M. Bramer, O. H. Franco, Levels of ambient air pollution according to mode of transport: a systematic review, The Lancet Public Health 2 (2017) e23–e34.

[5] V. Dheeraj Alshetty, S. K. Kuppili, S. M. Nagendra, G. Ramadurai, V. Sethi, R. Kumar, N. Sharma, A. Namdeo, M. Bell, P. Goodman, T. Chatterton, J. Barnes, L. De Vito, J. Longhurst, Characteristics of tail pipe (Nitric oxide) and resuspended dust emissions from urban roads – A case study in Delhi city, Journal of Transport and Health 17 (2020).

[6] K. J. Maji, A. Namdeo, D. Hoban, M. Bell, P. Goodman, S. S. Nagendra, J. Barnes, L. De Vito, E. Hayes, J. Longhurst, R. Kumar, N. Sharma, S. K. Kuppili, D. Alshetty, Analysis of various transport modes to evaluate personal exposure to PM2.5 pollution in Delhi, Atmospheric Pollution Research (2020). doi: 10.1016/j.apr.2020.12.003.

[7] WHO, Ambient (outdoor) air pollution, WHO, 2018. URL: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health. Accessed on: 24^{th} Sept, 2020

[8] S. C. Anenberg, J. Miller, D. K. Henze, R. Minjares, P. Achakulwisut, The global burden of transportation tailpipe emissions on air pollution-related mortality in 2010 and 2015, Environmental Research Letters 14 (2019) 094012.

[9] M. Guarnieri, Looking back to electric cars, in: 3rd Region-8 IEEE HISTory of Electro - Technology CONference: The Origins of Electrotechnologies, HISTELCON 2012 - Conference Proceedings, IEEE, 2012, pp. 1–6.

[10] M. Guarnieri, When cars went electric, Part 2, IEEE Industrial Electronics Magazine 5 (2011) 46–53.

[11] A. P. Loeb, Steam versus electric versus internal combustion: Choosing vehicle technol-

ogy at the start of the automotive age, Transportation Research Record (2004) 1–7.

[12] R. Matthé, U. Eberle, The Voltec System-Energy Storage and Electric Propulsion, in: Lithium-Ion Batteries: Advances and Applications, Elsevier, 2014, pp. 151–176.

[13] IEA, Global EV Outlook 2020, Technical Report, IEA, Paris, 2020.

[14] M. Woodward, J. Hamilton, B. Walton, J. Ringrow, G. Alberts, S. Fullerton-Smith, E. Day, Electric vehicles Setting a course for 2030, Deloitte Insights (2020). URL: https://www2.deloitte.com/uk/en/insights/focus/future-of-mobility/electric-vehicle-trends-2030.html. Accessed on: 24^{th} Sept, 2020

[15] T. Gersdorf, P. Hertzke, P. Schaufuss, S. Schenk, McKinsey Electric Vehicle Index: Europe cushions a global plunge in EV sales, McKinsey (2020). URL: https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/mckinsey-electric-vehicle-index-europe-cushions-a-global-plunge-in-ev-sales#. Accessed on: 24^{th} Sept, 2020

[16] S. P. Holland, E. T. Mansur, N. Z. Muller, A. J. Yates, Distributional Effects of Air Pollution from Electric Vehicle Adoption, Journal of the Association of Environmental and Resource Economists 6 (2019) S65–S94.

[17] W.-Y. Lin, M.-C. Hsiao, P.-C. Wu, J. S. Fu, L.-W. Lai, H.-C. Lai, Analysis of air quality and health co-benefits regarding electric vehicle promotion coupled with power plant emissions, Journal of Cleaner Production 247 (2020) 119152.

[18] T. R. Hawkins, B. Singh, G. Majeau-Bettez, A. H. Strømman, Comparative Environmental Life Cycle Assessment of Conventional and Electric Vehicles, Journal of Industrial Ecology 17 (2013) 53–64.

[19] Z. Yang, B. Wang, K. Jiao, Life cycle assessment of fuel cell, electric and internal combustion engine vehicles under different fuel scenarios and driving mileages in China, Energy 198 (2020) 117365.

[20] M. Messagie, F. S. Boureima, T. Coosemans, C. Macharis, J. V. Mierlo, A range-based vehicle life cycle assessment incorporating variability in the environmental assessment of different vehicle technologies and fuels, Energies 7 (2014) 1467–1482.

[21] A. Karpenko, T. Kinnunen, K. Framling, B. Dave, Open IoT ecosystem for smart EV charging, in: 2018 Global Internet of Things Summit, GIoTS 2018, IEEE, 2018, pp. 1–6.

[22] A. Karpenko, T. Kinnunen, M. Madhikermi, J. Robert, K. Främling, B. Dave, A. Nurminen, Data exchange interoperability in iot ecosystem for smart parking and EV charging, Sensors (Switzerland) 18 (2018) 4404.

[23] F. Patel, N. Saini, Breaking down EV Myths in India - Economics, The Cli-

mate Group 2020. URL: https://www.theclimategroup.org/our-work/news/breaking-down-ev-myths-india-economics. Accessed on: 30^{th} Sept, 2020.

[24] M. Sivak, B. Schoettle, Relative Costs of Driving Electric and Gasoline Vehicles in the Individual U.S. States, Technical Report, University of Michigan, 2018 1-31. URL: https://trid.trb.org/view/1508116.

[25] S. Brand, M. Petri, P. Haas, C. Krettek, C. Haasper, Hybrid and electric low-noise cars cause an increase in traffic accidents involving vulnerable road users in urban areas, International Journal of Injury Control and Safety Promotion 20 (2013) 339–341.

[26] E. Parizet, W. Ellermeier, R. Robart, Auditory warnings for electric vehicles: Detectability in normal-vision and visually-impaired listeners, Applied Acoustics 86 (2014) 50–58.

[27] P. H. L. Notten, D. L. Danilov, Battery Modeling: A Versatile Tool to Design Advanced Battery Management Systems, Advances in Chemical Engineering and Science 04 (2014) 62–72.

[28] Battery University, BU-205: Types of Lithium-ion, 2019. URL: https://batteryuniversity.com/learn/article/types_of_lithium_ion. Accessed on: 30^{th} Sept, 2020.

[29] Z. Chłopek, J. Lasocki, P. Wójcik, A. J. Badyda, Experimental investigation and comparison of energy consumption of electric and conventional vehicles due to the driving pattern, International Journal of Green Energy 15 (2018) 773–779.

[30] Y. Hübner, P. T. Blythe, G. A. Hill, M. Neaimeh, C. Higgins, Use of ITS to overcome barriers to the introduction of electric vehicles in the North East of England, 19th Intelligent Transport Systems World Congress, ITS 2012 (2012) 1–8.

[31] H. Zhang, X. Song, T. Xia, M. Yuan, Z. Fan, R. Shibasaki, Y. Liang, Battery electric vehicles in Japan: Human mobile behavior based adoption potential analysis and policy target response, Applied Energy 220 (2018) 527–535.

[32] T. Sakai, H. Miyamura, N. Kuriyama, I. Uehara, M. Muta, A. Takagi, U. Kajiyama, K. Kinoshita, F. Isogai, Nickel-metal hydride battery for electric vehicles, Journal of Alloys and Compounds 192 (1993) 158–160.

[33] T. Esaka, H. Sakaguchi, S. Kobayashi, Hydrogen storage in proton-conductive perovskite-type oxides and their application to nickel-hydrogen batteries, Solid State Ionics 166 (2004) 351–357.

[34] G. Deng, Y. Chen, M. Tao, C. Wu, X. Shen, H. Yang, M. Liu, Electrochemical properties and hydrogen storage mechanism of perovskite-type oxide LaFeO3 as a negative electrode for Ni/MH batteries, Electrochimica Acta 55 (2010) 1120–1124.

[35] L. Damen, J. Hassoun, M. Mastragostino, B. Scrosati, Solid-state, rechargeable Li/LiFePO4 polymer battery for electric vehicle application, Journal of Power Sources

195 (2010) 6902–6904.

[36] M. Kotobuki, Y. Suzuki, K. Kanamura, Y. Sato, K. Yamamoto, T. Yoshida, A novel structure of ceramics electrolyte for future lithium battery, Journal of Power Sources 196 (2011) 9815–9819.

[37] X. Xi, R. Sioshansi, V. Marano, Simulation-optimization model for location of a public electric vehicle charging infrastructure, Transportation Research Part D: Transport and Environment 22 (2013) 60–69.

[38] P. Morrissey, P. Weldon, M. O'Mahony, Future standard and fast charging infrastructure planning: An analysis of electric vehicle charging behaviour, Energy Policy 89 (2016) 257–270.

[39] M. Grote, J. Preston, T. Cherrett, N. Tuck, Locating residential on-street electric vehicle charging infrastructure: A practical methodology, Transportation Research Part D: Transport and Environment 74 (2019) 15–27.

[40] W. Shabbir, S. A. Evangelou, Real-time control strategy to maximize hybrid electric vehicle powertrain efficiency, Applied Energy 135 (2014) 512–522.

[41] Y. Guan, Z. Q. Zhu, I. A. Afinowi, J. C. Mipo, P. Farah, Difference in maximum torque-speed characteristics of induction machine between motor and generator operation modes for electric vehicle application, Electric Power Systems Research 136 (2016) 406–414.

[42] C. Chatzikomis, M. Zanchetta, P. Gruber, A. Sorniotti, B. Modic, T. Motaln, L. Blagotinsek, G. Gotovac, An energy-efficient torque-vectoring algorithm for electric vehicles with multiple motors, Mechanical Systems and Signal Processing 128 (2019) 655–673.

[43] J. C. Ferreira, V. Monteiro, J. L. Afonso, Data mining approach for range prediction of electric vehicle, Conference on future automotive technology - Focus electromobility (2012) pp. 1–15.

[44] S. C. Yang, M. Li, Y. Lin, T. Q. Tang, Electric vehicle's electricity consumption on a road with different slope, Physica A: Statistical Mechanics and its Applications 402 (2014) 41–48.

[45] P. Ondrúška, I. Posner, The route not taken: Driver-centric estimation of electric vehicle range, Proceedings International Conference on Automated Planning and Scheduling, ICAPS 2014-Janua (2014) 413–420.

[46] OpenWeatherMap, Weather API from Open Weather Map, 2020. URL: `https://openweathermap.org/api`. Accessed on: 31^{st} Nov, 2019.

[47] H. Zhai, H. C. Frey, N. M. Rouphail, A vehicle-specific power approach to speed- and facility-specific emissions estimates for diesel transit buses, Environmental Science and Technology 42 (2008) 7985–7991.

[48] P. Sharer, Powertrain Systems Analysis Toolkit (PSAT), in: Proceedings of The PHEV

2007 conference : where the grid meets the road, Winnipeg, MB (Canada), 2007, pp. 1–43.

[49] K. B. Wipke, M. R. Cuddy, S. D. Burch, ADVISOR 2.1: A user-friendly advanced powertrain simulation using a combined backward/forward approach, IEEE Transactions on Vehicular Technology 48 (1999) 1751–1761.

[50] A. Brooker, J. Gonder, L. Wang, E. Wood, S. Lopp, L. Ramroth, FASTSim: A Model to Estimate Vehicle Efficiency, Cost and Performance, in: SAE 2015 World Congress & Exhibition, 2015. doi: 10.4271/2015-01-0973.

[51] Powersim Inc., PSIM Website, 2020. URL: https://powersimtech.com/. Accessed on: 31^{st} Nov, 2020.

[52] D. W. Gao, C. Mi, A. Emadi, Modeling and simulation of electric and hybrid vehicles, Proceedings of the IEEE 95 (2007) 729–745.

[53] K. N. Genikomsakis, G. Mitrentsis, A computationally efficient simulation model for estimating energy consumption of electric vehicles in the context of route planning applications, Transportation Research Part D: Transport and Environment 50 (2017) 98–118.

[54] T. Halmeaho, P. Rahkola, K. Tammi, J. Pippuri, A. P. Pellikka, A. Manninen, S. Ruotsalainen, Experimental validation of electric bus powertrain model under city driving cycles, IET Electrical Systems in Transportation 7 (2017) 74–83.

[55] A. Donkers, D. Yang, M. Viktorović, Influence of driving style, infrastructure, weather and traffic on electric vehicle performance, Transportation Research Part D: Transport and Environment 88 (2020) 102569.

[56] Argonne National Laboratory Transportation Technology R&D Center, Downloadable Dynamometer Database, 2015. URL: http://www.transportation.anl.gov/D3/. Accessed on: 3^{rd} Oct, 2018.

[57] I. Miri, A. Fotouhi, N. Ewin, Electric vehicle energy consumption modelling and estimation—A case study, International Journal of Energy Research (2020). doi: 10.1002/er.5700.

[58] X. Wu, D. Freese, A. Cabrera, W. A. Kitch, Electric vehicles' energy consumption measurement and estimation, Transportation Research Part D: Transport and Environment 34 (2015) 52–67.

[59] R. Maia, M. Silva, R. Araújo, U. Nunes, Electrical vehicle modeling: A fuzzy logic model for regenerative braking, Expert Systems with Applications 42 (2015) 8504–8519.

[60] C. Fiori, K. Ahn, H. A. Rakha, Power-based electric vehicle energy consumption model: Model development and validation, Applied Energy 168 (2016) 257–268.

[61] D. Jiménez, S. Hernández, J. Fraile-Ardanuy, J. Serrano, R. Fernández, F. Alvarez, Modelling the effect of driving events on electrical vehicle energy consumption using inertial

sensors in smartphones, Energies 11 (2018) 412.

[62] B. Luin, S. Petelin, F. Al-Mansour, Microsimulation of electric vehicle energy consumption, Energy 174 (2019) 24–32.

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

[64] K. Liu, J. Wang, T. Yamamoto, T. Morikawa, Modelling the multilevel structure and mixed effects of the factors influencing the energy consumption of electric vehicles, Applied Energy 183 (2016) 1351–1360.

[65] C. De Cauwer, W. Verbeke, T. Coosemans, S. Faid, J. Van Mierlo, A data-driven method for energy consumption prediction and energy-efficient routing of electric vehicles in real-world conditions, Energies 10 (2017).

[66] R. Galvin, Energy consumption effects of speed and acceleration in electric vehicles: Laboratory case studies and implications for drivers and policymakers, Transportation Research Part D: Transport and Environment 53 (2017) 234–248.

[67] G. M. Fetene, S. Kaplan, S. L. Mabit, A. F. Jensen, C. G. Prato, Harnessing big data for estimating the energy consumption and driving range of electric vehicles, Transportation Research Part D: Transport and Environment 54 (2017) 1–11.

[68] X. Yuan, C. Zhang, G. Hong, X. Huang, L. Li, Method for evaluating the real-world driving energy consumptions of electric vehicles, Energy 141 (2017) 1955–1968.

[69] K. Liu, T. Yamamoto, T. Morikawa, Impact of road gradient on energy consumption of electric vehicles, Transportation Research Part D: Transport and Environment 54 (2017) 74–81.

[70] X. Qi, G. Wu, K. Boriboonsomsin, M. J. Barth, Data-driven decomposition analysis and estimation of link-level electric vehicle energy consumption under real-world traffic conditions, Transportation Research Part D: Transport and Environment 64 (2018) 36–52.

[71] F. C. López, R. Á. Fernández, Predictive model for energy consumption of battery electric vehicle with consideration of self-uncertainty route factors, Journal of Cleaner Production 276 (2020) 124188.

[72] K. Liu, J. Wang, T. Yamamoto, T. Morikawa, Exploring the interactive effects of ambient temperature and vehicle auxiliary loads on electric vehicle energy consumption, Applied Energy 227 (2018) 324–331.

[73] Y. Xu, K. Wang, Research on Estimation Method of Mileage Power Consumption for Electric Vehicles, in: Proceedings of the 2018 International Conference on Computer Science, Electronics and Communication Engineering (CSECE 2018), Atlantis Press, Paris, France, 2018.

[74] C. Lv, X. Hu, A. Sangiovanni-Vincentelli, Y. Li, C. M. Martinez, D. Cao, Driving-Style-

Based Codesign Optimization of an Automated Electric Vehicle: A Cyber-Physical System Approach, IEEE Transactions on Industrial Electronics 66 (2019) 2965–2975.

[75] A. Diaz Alvarez, F. Serradilla Garcia, J. E. Naranjo, J. J. Anaya, F. Jimenez, Modeling the driving behavior of electric vehicles using smartphones and neural networks, IEEE Intelligent Transportation Systems Magazine 6 (2014) 44–53.

[76] J. Felipe, J. C. Amarillo, J. E. Naranjo, F. Serradilla, A. Diaz, Energy Consumption Estimation in Electric Vehicles Considering Driving Style, IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC 2015-Octob (2015) 101–106.

[77] J. Zhang, Z. Wang, P. Liu, Z. Zhang, Energy consumption analysis and prediction of electric vehicles based on real-world driving data, Applied Energy 275 (2020) 115408.

[78] E. Kalapanidas, N. Avouris, M. Craciun, D. Neagu, Machine Learning Algorithms: A study on noise sensitivity, in: Proc. 1st Balcan Conference in Informatics, October, 2003, pp. 356–365.

[79] M. Levin, Y.-D. Tsao, On forecasting freeway occupancies and volumes, Transportation Research Record (1980) 47–49.

[80] M. M. Hamed, H. R. Ai-Masaeid, Z. M. Bani Said, Short-term prediction of traffic volume in urban arterials, Journal of Transportation Engineering 121 (1995) 249–254.

[81] B. M. Williams, Multivariate vehicular traffic flow prediction: Evaluation of ARIMAX modeling, Transportation Research Record 1776 (2001) 194–200.

[82] B. M. Williams, L. A. Hoel, Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results, Journal of Transportation Engineering 129 (2003) 664–672.

[83] S. V. Kumar, L. Vanajakshi, Short-term traffic flow prediction using seasonal ARIMA model with limited input data, European Transport Research Review 7 (2015) 21.

[84] S. R. Chandra, H. Al-Deek, Predictions of freeway traffic speeds and volumes using vector autoregressive models, Journal of Intelligent Transportation Systems: Technology, Planning, and Operations 13 (2009) 53–72.

[85] I. Okutani, Y. J. Stephanedes, Dynamic prediction of traffic volume through Kalman filtering theory, Transportation Research Part B 18 (1984) 1–11.

[86] Y. Xie, Y. Zhang, Z. Ye, Short-term traffic volume forecasting using Kalman filter with discrete wavelet decomposition, Computer-Aided Civil and Infrastructure Engineering 22 (2007) 326–334.

[87] L. L. Ojeda, A. Y. Kibangou, C. C. De Wit, Adaptive Kalman filtering for multi-step ahead traffic flow prediction, in: Proceedings of the American Control Conference, IEEE, 2013, pp. 4724–4729.

[88] P. K. Mall, P. K. Singh, D. Yadav, GLCM based feature extraction and medical X-RAY image classification using machine learning techniques, in: 2019 IEEE Conference on

Information and Communication Technology, CICT 2019, IEEE, 2019, pp. 1–6.

[89] Y. Zhang, L. Wu, Weights optimization of neural network via improved BCO approach, Progress in Electromagnetics Research 83 (2008) 185–198.

[90] G. A. Davis, N. L. Nihan, Nonparametric Regression and Short-Term Freeway Traffic Forecasting, Journal of Transportation Engineering 117 (1991) 178–188.

[91] H. Chang, Y. Lee, B. Yoon, S. Baek, Dynamic near-term traffic flow prediction: system-oriented approach based on past experiences, IET Intelligent Transport Systems 6 (2012) 292.

[92] C. H. Wu, J. M. Ho, D. T. Lee, Travel-time prediction with support vector regression, IEEE Transactions on Intelligent Transportation Systems 5 (2004) 276–281.

[93] H. Su, L. Zhang, S. Yu, Short-term traffic flow prediction based on incremental support vector regression, Proceedings - Third International Conference on Natural Computation, ICNC 2007 1 (2007) 640–645.

[94] M. Castro-Neto, Y. S. Jeong, M. K. Jeong, L. D. Han, Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions, Expert Systems with Applications 36 (2009) 6164–6173.

[95] A. Khotanzad, N. Sadek, Multi-scale high-speed network traffic prediction using combination of neural networks, in: Proceedings of the International Joint Conference on Neural Networks, volume 2, IEEE, 2003, pp. 1071–1075.

[96] A. Csikós, Z. J. Viharos, K. B. Kis, T. Tettamanti, I. Varga, Traffic speed prediction method for urban networks - An ANN approach, in: 2015 International Conference on Models and Technologies for Intelligent Transportation Systems, MT-ITS 2015, IEEE, 2015, pp. 102–108.

[97] S. Sun, C. Zhang, G. Yu, A Bayesian network approach to traffic flow forecasting, IEEE Transactions on Intelligent Transportation Systems 7 (2006) 124–133.

[98] E. Castillo, J. M. Menéndez, S. Sánchez-Cambronero, Predicting traffic flow using Bayesian networks, Transportation Research Part B: Methodological 42 (2008) 482–509.

[99] X. Dong, T. Lei, S. Jin, Z. Hou, Short-term traffic flow prediction based on XGBoost, in: Proceedings of 2018 IEEE 7th Data Driven Control and Learning Systems Conference, DDCLS 2018, 2018, pp. 854–859.

[100] M. Zhang, X. Fei, Z. hui Liu, Short-term traffic flow prediction based on combination model of Xgboost-LightGBM, in: Proceedings - 2018 International Conference on Sensor Networks and Signal Processing, SNSP 2018, IEEE, 2019, pp. 322–327.

[101] N. Zarei, M. A. Ghayour, S. Hashemi, Road traffic prediction using context-aware random forest based on volatility nature of traffic flows, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes

in Bioinformatics), volume 7802 LNAI, 2013, pp. 196–205.

[102] Y. Liu, H. Wu, Prediction of road traffic congestion based on random forest, in: Proceedings - 2017 10th International Symposium on Computational Intelligence and Design, ISCID 2017, volume 2, IEEE, 2018, pp. 361–364.

[103] S. Zhang, Y. Yao, J. J. Hu, Y. Zhao, S. Li, J. J. Hu, Deep autoencoder neural networks for short-term traffic congestion prediction of transportation networks, Sensors (Switzerland) 19 (2019).

[104] X. Ma, Z. Dai, Z. He, J. Ma, Y. Y. Y. Wang, Y. Y. Y. Wang, Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction, Sensors (Switzerland) 17 (2017).

[105] L. Li, L. Qin, X. Qu, J. Zhang, Y. Wang, B. Ran, Day-ahead traffic flow forecasting based on a deep belief network optimized by the multi-objective particle swarm algorithm, Knowledge-Based Systems 172 (2019) 1–14.

[106] Z. Cui, R. Ke, Y. Wang, Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction, CoRR abs/1801.0 (2018).

[107] D. Liu, L. Tang, G. Shen, X. Han, Traffic speed prediction: An attention-based method, Sensors (Switzerland) 19 (2019).

[108] J. Liu, Z.-J. Zha, X. Chen, Z. Wang, Y. Zhang, Dense 3D-Convolutional Neural Network for Person Re-Identification in Videos, ACM Transactions on Multimedia Computing, Communications, and Applications 15 (2019) 1–19.

[109] S. Masood, A. Rai, A. Aggarwal, M. N. Doja, M. Ahmad, Detecting distraction of drivers using Convolutional Neural Network, Pattern Recognition Letters (2018). doi:10.1016/j.patrec.2017.12.023.

[110] Y. Wu, H. Tan, L. Qin, B. Ran, Z. Jiang, A hybrid deep learning based traffic flow prediction method and its understanding, Transportation Research Part C: Emerging Technologies 90 (2018) 166–180.

[111] L. Geng, J. Sun, Z. Xiao, F. Zhang, J. Wu, Combining CNN and MRF for road detection, Computers & Electrical Engineering 70 (2018) 895–903.

[112] X. Song, T. Rui, S. Zhang, J. Fei, X. Wang, A road segmentation method based on the deep auto-encoder with supervised learning, Computers & Electrical Engineering 68 (2018) 381–388.

[113] R. Nayak, U. C. Pati, S. K. Das, A comprehensive review on deep learning-based methods for video anomaly detection, Image and Vision Computing (2020) 104078.

[114] L. De Bortoli, F. Guzzi, S. Marsi, S. Carrato, G. Ramponi, A Fast Face Recognition CNN Obtained by Distillation, in: Lecture Notes in Electrical Engineering, volume 627, 2020, pp. 341–347.

[115] F. Guzzi, L. De Bortoli, S. Marsi, S. Carrato, G. Ramponi, Distillation of a CNN for

a high accuracy mobile face recognition system, in: 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2019 - Proceedings, IEEE, 2019, pp. 989–994.

[116] U.S. Department of Energy, 2013 Nissan Leaf Advanced Vehicle Testing - Baseline Testing Results, Technical Report, 2013.

[117] T. Burress, Benchmarking State-of-the-Art Technologies, Technical Report, Oak Ridge National Laboratory, 2013. URL: http://energy.gov/sites/prod/files/2014/03/f13/ape006_burress_2013_o.pdf.

[118] J. B. Yang, M. N. Nguyen, P. P. San, X. L. Li, S. Krishnaswamy, Deep convolutional neural networks on multichannel time series for human activity recognition, IJCAI International Joint Conference on Artificial Intelligence 2015-January (2015) 3995–4001.

[119] Z. Wang, T. Oates, Encoding time series as images for visual inspection and classification using tiled convolutional neural networks, AAAI Workshop - Technical Report WS-15-14 (2015) 40–46.

[120] J. C. B. Gamboa, Deep Learning for Time-Series Analysis (2017). URL: http://arxiv.org/abs/1701.01887.

[121] T. Semwal, G. Mathur, P. Yenigalla, S. B. Nair, A practitioners' guide to transfer learning for text classification using convolutional neural networks, in: SIAM International Conference on Data Mining, SDM 2018, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2018, pp. 513–521.

[122] B. Biswas, S. K. Ghosh, A. Ghosh, C. Chakraborty, P. Mitra, Target Object Recognition Using Multiresolution SVD and Guided Filter with Convolutional Neural Network, International Journal of Pattern Recognition and Artificial Intelligence (2020). doi: 10.1142/S0218001420520084.

[123] A. Krizhevsky, I. Sutskever, G. E. Hinton, ImageNet classification with deep convolutional neural networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Communications of the ACM, volume 60, Curran Associates, Inc., 2017, pp. 84–90.

[124] A. P. Twinanda, S. Shehata, D. Mutter, J. Marescaux, M. De Mathelin, N. Padoy, EndoNet: A Deep Architecture for Recognition Tasks on Laparoscopic Videos, IEEE Transactions on Medical Imaging 36 (2017) 86–97.

[125] R. Shwartz-Ziv, N. Tishby, Opening the Black Box of Deep Neural Networks via Information (2017) 1–19.

[126] G. Devineau, F. Moutarde, W. Xi, J. Yang, 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.

[127] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in:

Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, volume 2016-Decem, IEEE, 2016, pp. 770–778.

[128] Y. D. Zhang, C. Pan, J. Sun, C. Tang, Multiple sclerosis identification by convolutional neural network with dropout and parametric ReLU, Journal of Computational Science 28 (2018) 1–10.

[129] L. Breiman, Bagging predictors, Machine Learning 24 (1996) 123–140.

[130] N. Kularatna, 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.

[131] Met Office, National Meteorological Library and Archive Fact sheet No. 6 – The Beaufort Scale, Technical Report, 2007.

[132] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Journal of Machine Learning Research, volume 9, 2010, pp. 249–256.

[133] D. P. Kingma, J. L. Ba, Adam: A method for stochastic optimization, 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2015). URL: http://arxiv.org/abs/1412.6980.

[134] K. S. Sang, B. Zhou, P. Yang, Z. Yang, A Survey on Urban Traffic Optimisation for Sustainable and Resilient Transportation Network, in: Proceedings - 2016 9th International Conference on Developments in eSystems Engineering, DeSE 2016, IEEE, 2017, pp. 233–238.

[135] Y. Lin, P. Wang, M. Ma, Intelligent Transportation System(ITS): Concept, Challenge and Opportunity, in: Proceedings - 3rd IEEE International Conference on Big Data Security on Cloud, BigDataSecurity 2017, 3rd IEEE International Conference on High Performance and Smart Computing, HPSC 2017 and 2nd IEEE International Conference on Intelligent Data and Securit, IEEE, 2017, pp. 167–172.

[136] M. Bode, S. S. Jha, S. B. Nair, On Movement of Emergency Services amidst Urban Traffic, ICST Transactions on Ambient Systems 2 (2015) 150713.

[137] J. O'Brien, A. Namdeo, M. Bell, P. Goodman, A congestion sensitive approach to modelling road networks for air quality management, International Journal of Environment and Pollution 54 (2014) 213–221.

[138] L. I.-K. Lin, A Concordance Correlation Coefficient to Evaluate Reproducibility, Biometrics 45 (1989) 255.

[139] M. Behzadian, S. Khanmohammadi Otaghsara, M. Yazdani, J. Ignatius, A state-of the-art survey of TOPSIS applications, Expert Systems with Applications 39 (2012) 13051–13069.

[140] R. Nayak, U. C. Pati, S. K. Das, Video Anomaly Detection using Convolutional Spatiotemporal Autoencoder, in: 2020 International Conference on Contemporary Comput-

ing and Applications, IC3A 2020, IEEE, 2020, pp. 175–180.

[141] M. Saha, A. Santara, P. Mitra, A. Chakraborty, R. S. Nanjundiah, Prediction of the Indian summer monsoon using a stacked autoencoder and ensemble regression model, International Journal of Forecasting 37 (2021) 58–71.

[142] P. Vincent, H. Larochelle, Y. Bengio, P. A. Manzagol, 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.

[143] J. Masci, U. Meier, D. Cireşan, J. Schmidhuber, 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), volume 6791 LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 52–59.

[144] California Department of Transportation, Caltrans Performance Measurement System (PeMS) – State of California, 2020. URL: http://pems.dot.ca.gov/. Accessed on: 4^{th} Jan, 2020

[145] B. Sharma, S. Kumar, P. Tiwari, P. Yadav, M. I. Nezhurina, ANN based short-term traffic flow forecasting in undivided two lane highway, Journal of Big Data 5 (2018) 48.

[146] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, J. Sun, A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting, Transportation Research Part C: Emerging Technologies 62 (2016) 21–34.

[147] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, volume 13-17-Augu of *KDD '16*, Association for Computing Machinery, New York, NY, USA, 2016, pp. 785–794.

[148] Y. Gu, W. Lu, L. Qin, M. Li, Z. Shao, Short-term prediction of lane-level traffic speeds: A fusion deep learning model, Transportation Research Part C: Emerging Technologies 106 (2019) 1–16.

[149] The Heidelberg Institute for Geoinformation Technology, Openroute Service, 2020. URL: https://openrouteservice.org/. Accessed on: 4^{th} Jan, 2020

[150] Meteostat, Meteostat, 2020. URL: https://dev.meteostat.net/. Accessed on: 4^{th} Jan, 2020

List of Publications

1. S. Modi, J. Bhattacharya, and P. Basak, "Estimation of energy consumption of electric vehicles using Deep Convolutional Neural Network to reduce driver's range anxiety," *ISA Transactions*, vol. 98, pp. 454–470, 2020.
   DOI: https://doi.org/10.1016/j.isatra.2019.08.055
   SCI Impact Factor: 4.305

2. S. Modi, J. Bhattacharya, and P. Basak, "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 Computing*, vol. 25, no. 3, pp. 2399–2416, 2021.
   DOI: https://doi.org/10.1007/s00500-020-05310-y
   SCI Impact Factor: 3.050

3. S. Modi, J. Bhattacharya, and P. Basak, "Multistep traffic speed prediction: A deep learning based approach using latent space mapping considering spatio-temporal dependencies," *Expert Systems with Applications*.
   SCI Impact Factor: 5.452 (Under review)