Journal of Signal Processing Systems (2023) 95:161–176
https://doi.org/10.1007/s11265-022-01831-x

LSTM Network Integrated with Particle Filter for Predicting the Bus Passenger Traffic

G S Vidya[1,2] · **V S Hari**[3]

Received: 23 June 2022 / Revised: 22 December 2022 / Accepted: 24 December 2022 / Published online: 12 January 2023

Abstract
The paper reports a combination of the deep learning technique and bayesian filtering to effectively predict the passenger traffic. The architecture of the model integrates the particle filter with the LSTM network. The time series sequential prediction is best achieved using LSTM network while Markovian behaviour is well extracted using Bayesian (Particle Filter) filters. The temporal and spatial features of the traffic data are analyzed. Three relevant temporal variations *viz.*, morning, noon and post noon patterns are identified after the histogram analysis. These patterns are statistically modelled and the integrated model is used to accurately predict the passenger flow for the next thirty days, facilitating, the bus scheduling for that period. The experimental results proved that the proposed integrated model with coefficient of determination (R^2) value of 0.88 is functional in predicting the passenger traffic even when the training data set size is small.

Keywords Deep Learning · Bayesian filters · LSTM · Particle filter · Markov chain · R^2 value

1 Introduction

Public transportation is the mode of travel for masses. The enhancement in services, provided by transport systems, are beneficial to common man. Technologies are being developed to improve passenger comfort while enhancing the profit of service providers. Automation, based on artificial intelligence and deep learning, when employed in passenger data analysis will facilitate the scheduling of buses, based on passenger demand. The passenger data flow is complex and random in nature, influenced by many fixed and stochastic parameters. The passenger data prediction is a time series prediction problem in which random dependencies are embedded, which is more complex than normal predictive modelling. The great demand for accurate passenger prediction motivated a large amount of research work in this field. non parametric machine learning models like Gaussian Process Regression [1] was used to predict the passenger data, modelling the passenger arrival as a poisson process. Parametric ML models are further developed to obtain more prediction accuracy. Several statistical and neural network models are being used, as detailed in the next section. The passenger traffic is a sequential time series data exhibiting nongaussian and nonlinear nature. The sequential data prediction is best achieved using an LSTM network. Three temporal patterns are extracted and seperate LSTM models are used for prediction. Thus the passenger flow prediction is the need of the hour. This leads to automated crew scheduling, which in turn, leads to intelligent transport systems. Such systems are envisaged to improve the punctuality of services, comfort of the passengers and profit of public transport systems.

The passenger traffic and the scheduling process are stochastic in nature. AI and Machine learning models work better for modelling such stochastic dynamic systems. Future lies in intelligent transport systems that adapt to passenger needs and road conditions. Transport companies [2] are now using latest technologies for automating all systems to improve the passenger comfort as well as to improve the earnings per km. The machine learning models work better for large datasets. The dataset used here is a nine months data. Even for this data, LSTM network supported by particle filter is proposed

✉ G S Vidya
vidyahari.gs@gmail.com

V S Hari
harivs@ceconline.edu

[1] Department of Electronics, College of Engineering Chengannur, Kerala, India 691521

[2] A P J Abdul Kalam Technological University, Thiruvananthapuram, India

[3] Department of Electronics, College of Engineering, Chengannur, Kerala, India 691521

that overcomes the difficulty of limited dataset. The paper proposes a hybrid method to correct the LSTM output to obtain more accurate prediction results. The detailed analysis of passenger traffic exhibits Markovian behaviour, which necessitated the use of a nonlinear filter that can extract and model Markov property exhibited by them . So to achieve better prediction results, LSTM coupled with particle filters [3] is used as a prediction model. This work is explained in detail in the following sections.

2 Literature Survey

The short term prediction of passenger traffic started in the late seventies with the work of [4], which states the application of Autoregressive Integrated Moving Average(ARIMA) model. The studies by [5] reveals that the temporal accumulation of data would alter the features and trends of the data. It is essential to accurately predict the distribution of passenger flow to ensure efficient operation management. The neural network based integrated models always outperform simple statistical models, in modelling multidimensional stochastic data. Passenger travel flow during holidays show distinct characteristics different from normal days, such data is extracted and efficiently managed, and used a forecasting distribution theory based on Neural Network [5]. Ge et al. proposed trend moving average method as a rational and effective method in predicting the bus passenger traffic [6].

A stochastic process is a mathematical tool that describes a random phenomenon evolving in time. Markov chains are among the most important stochastic processes. They are stochastic processes for which the description of the present state fully captures all the information that could influence the future evolution of the process. Predicting traffic flows, simulation of efficient road traffic model [7] communications networks, genetic issues, and queues are examples where Markov chains can be used to model performance [8]. The application of Markov chains to analyze the behaviour of complex systems is well known in several fields, like, subsurface characterization [9], signal processing [10], finance [11], analysing the spatial distribution of heterogeneous vehicle headways in mixed traffic [12] etc. Recently hybrid technologies are used in many areas like communication [13], city pollution analysis [14] and distotion detection of Lidar signals [15]

The machine learning techniques like Random Forest and LSTM [16] are employed to predict the number of passengers entering each station or boarding at each stop. Neural Network based systems [17] and fuzzy based systems [18] are used in almost all area to improve the system performance. A special kind of scheme in deep learning, namely LSTM network [19] is used to effectively overcome the issues in the time series data [20]. An improved STL-LSTM model [21] is used for bus passenger traffic during the covid-19 pandemic situation. Another multi step prediction, based on Kalman Filter [22] and support vector regression method, is used for bus passenger load prediction [23]. These studies proved that a two step or an integrated model is more effective in the traffic prediction process. This paper proposes an augmentation to the LSTM network with a particle filter [24] to counteract the prediction errors due to small training dataset.

3 Theory

The time series sequential prediction of bus passenger traffic is best achieved using LSTM network. The passenger data is a Markovian process and Bayesian techniques are incorporated for their study. Thus the integrated model is based on modelling the passenger traffic as a discrete time Markov chain.

3.1 Markovian Model

A Markov chain is a stochastic process that satisfies the Markov property, which means that the future state depends only on the current state and, not on the previous events. This means that the current state of the process is only required to make the best possible prediction of its future. It often happens that the bus leaves a terminal with fully occupied passengers and reaches another terminal, with all the passengers alighting there, with the transition from one state to other depending only on the present sate and not on the previous history. This is the rationale for choosing a Markovian model. Considering the bus passenger traffic as a stochastic process

$$X = \{X_n, n \in N\} \tag{1}$$

in a countable space S is a discrete time Markov chain if $X_n \in S$, $\forall n \geq 0$ (where N represents the discrete time step).

For the Markovian traffic data, the transition probability is $P\{X_n = i_n | X_{n-1} = i_{n-1}\}$ such that

$$P\{X_n = i_n | X_{n-1} = i_{n-1}, \cdots, X_0 = i_0\} = P\{X_n = i_n | X_{n-1} = i_{n-1}\} \tag{2}$$

such that $i_0, \cdots i_{n-1}$, $\forall n \geq 1$. The transition probability matrix represents the likelihood of occupancy of bus in each route. The prediction process using Bayesian technique assigns a prior distribution for the transition probability matrix $\mathbf{P}$. The prior is assigned on each p_{ij} with the constraint that $\sum p_{ij} = 1$. The prior chosen is the Dirichlet prior as it is the conjugate prior of many probability distributions. The Dirichlet distribution is a multivariate distribution whose probability density function is given by

$$f(x_x, \ldots, x_k; \alpha_1, \cdots, \alpha_k) = \frac{1}{B(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1} \tag{3}$$

where $B(\alpha) = \frac{\prod_{i=1}^{k}\Gamma(\alpha)}{\Gamma(\sum_{i=1}^{k}\alpha_i)}$ and $\alpha = (\alpha_1, \cdots, \alpha_k)$ are the parameters of this distribution. This distribution is used in the particle filtering for prediction as discussed in Sec. 5.4.1.

3.2 Long Short Term Memory (LSTM) Network

Neural Networks are prototypes of our biological neural system, that learn and remember from the previous data, and predict values for new datasets. Recurrent Neural Network (RNN) is different from the normal network in a way that, it depends on the previous data to predict the upcoming data. This makes it unique to use them for sequential data problems [25]. RNNs have a unique state or feedback layer to store the output for a given data which is again used as input and hence the name recurrent. But the RNN networks face long term dependancy problem, that is efficiently corrected by LSTM networks.

LSTMs use a series of gates to control the flow of data. The typical structure of LSTM includes four gates, *viz.* Cell state gate, forget gate, input gate and output gate as shown in Fig. 1. Each gate has its own activation function and fully connected layer. The cell state gate z remembers the information of the previous data over time, and the forget gate z_f chooses the necessary data bits from previous data. The input gate, z_i is a sigmoid activated network which chooses the required content from the input data that output a vector of values in [0, 1] (due to the sigmoid activation). The output gate z_o selects the required data to be used for computing the output. It can be seen from Fig. 1 that, there are three inputs to an LSTM block, *viz* cell state c_{t-1}, previous hidden state h_{t-1}, and the current input x_t. The outputs of the LSTM block are cell state c_t, hidden state h_t, and current output y_t. The mathematical model of the LSTM units is as follows:

$$Z_f = \sigma\left(W_f[x_t, h_{t-1}]\right) \tag{4a}$$

$$Z_i = \sigma\left(W_i[x_t, h_{t-1}]\right) \tag{4b}$$

$$Z_o = \sigma\left(W_o[x_t, h_{t-1}]\right) \tag{4c}$$

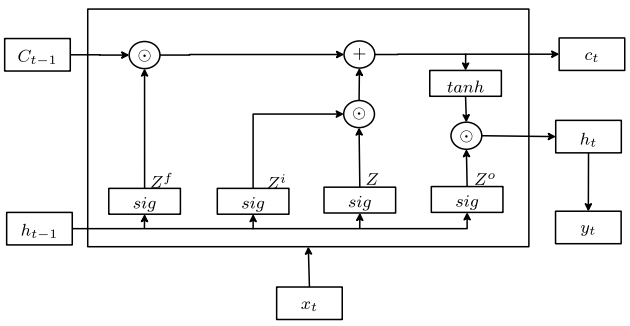


Figure 1 Structure of LSTM block.

$$c_t = Z_f \odot c_{t-1} + Z_i \odot \tanh(W[x_t, h_{t-1}]) \tag{4d}$$

$$h_t = Z_o \odot \tanh(c_t) \tag{4e}$$

$$y_t = \sigma(W_o \times h_t) \tag{4f}$$

where W_* indicates the weight matrix of the corresponding gates and $\odot$ is the Hadamard product and σ, the tan sigmoid activation function. The nonlinear activation functions are tanh and sigmoid, given by

$$S(x) = \frac{1}{1 + e^{-x}} \tag{5a}$$

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{5b}$$

The weight matrices determine the importance to accord to both the present input and the past hidden state. The weights are repetitively adjusted, using backpropagation algorithm, until the error becomes minimum.

3.3 Particle Filter

Particle filter is used for modelling nonlinear and nongaussian systems. The basic idea of particle filtering is the sequential Monte Carlo methodology that recursively computes the probability distributions using sequential importance sampling. The goal is to recursively estimate a state vector x, specifically, the hidden state sequence x_k of a dynamical system, whose initial state is u_k and $k \in N$ is the discrete time step. The hidden state means that no direct state measurements are available. The estimation process employs two models,

1. A process model that encodes prior knowledge on how the state x_k is expected to evolve over time.
2. A measurement model that relates measurements to the state x_k.

These two sources of information are modelled using nonlinear mathematical equations. The process model reflects the state changes over time, given noise and optional inputs:

$$x_k = f_k(x_{k-1}, u_k, v_{k-1}) \tag{6}$$

where f_k is a function that uniquely associates the state at time step $k-1$ with a state at time step k. The process model noise sequence v_{k-1} is independent and identically distributed that represents uncertainties related to the process model. The second source of information is the measurement model:

$$z_k = h_k(x_k, u_k, n_k) \tag{7}$$

where h_k is a function that associates the state with an expected measurement. Here n_k is an independent and identically distributed noise sequence representing the measurement noise.

Bayes theorem is used to refine the belief based on a prior estimate and newly received measurements. According to Bayesian perspective, the state estimate is represented by a posterior probability density function that quantifies both the estimated state and the uncertainty associated with the estimated value. The estimate of the state at time k given all measurements and inputs upto time k is denoted by the conditional pdf, $P(x_k|u_{1:k}, z_{1:k})$, where $u_{1:k} = u_i; i = 1, \ldots, k$ denotes the sequence of known control inputs and $z_{1:k} = z_i; i = 1, \ldots, k$ denotes the measurement sequence. The posterior state is the state that is estimated at each time step. The state sequence is modelled by a Markov chain, which implies that the past is adequately summarized by only the state at the previous time step. The particle filter is a Bayesian filter, in which estimation of a Markovian chain is performed using Bayesian theory. By Bayes theorem

$$\text{posterior} = \frac{\text{prior} \times \text{likelihood}}{\text{marginal likelihood}} \tag{8}$$

The marginal likelihood is the normalization term and depends only on measurements. The posterior distribution at the previous time step, $P(x_{k-1}|z_{1:k-1})$, is combined with the process model to form the prior state (graphically represented in Fig. 2):

$$P(x_k|z_{1:k-1}) = \int P(x_k|x_{k-1})P(x_{k-1}|z_{1:k-1})\, dx_{k-1} \tag{9}$$

The prior represents the predicted state at time k given measurements up to time $k-1$. During the update step, the measurement z_k at time k is used to compute the posterior using Bayes' theorem:

$$P(x_k|z_{1:k}) = \frac{P(z_k|x_k)P(x_k|z_{1:k-1})}{P(z_k|z_{1:k-1})} \tag{10}$$

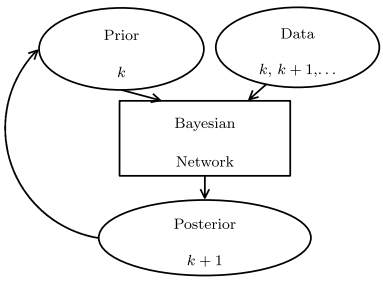


Figure 2 Bayesian filtering algorithm.

The likelihood $P(z_k|x_k)$ represents the conditional probability of a measurement given the predicted state, $P(x_k|z_{1:k-1})$ is the prior computed using Eq. 9 and the normalizing constant represents the probability of the measurement. It can be computed using:

$$P(z_k|z_{1:k-1}) = \int P(z_k|x_k)P(x_k|z_{1:k-1})\, dx_k \tag{11}$$

The posterior density and normalizing constant, described by Eqs. 10 and 11, are implemented in particle filter by representing them using discrete pdf. The optimal Bayesian solution is given by a sum of weighted samples:

$$P(x_{0:k}|z_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \delta(x_{0:k} - x_{0:k}^i) \tag{12}$$

Here $\{w_k^i, x_{0:k}^i\}_{i=1}^{N_s}$ is a set containing N_s samples and weights, in which each sample $x_{0:k}^i$ represents realization of the state sequence. These samples are named as particles. The weight w_k^i represents the relative importance of each of the N_s samples $x_{0:k}^i$ and $\sum_{i=1}^{N_s} w_k^i = 1$. Samples with high weights are closer to the true state sequence than samples associated with low weights. The Dirac delta function $\delta(a)$ is zero everywhere except a, with its integral being unity. The advantages of representing the posterior by a set of weighted particles such as

1. the ability to represent arbitrarily shaped pdfs and
2. minimal restrictions on the process and measurement models.

are the main reasons for the popularity of particle filter.

The posterior pdf that must be estimated is unknown, making sampling impossible. Samples must therefore be drawn from another distribution instead. This distribution is known as importance density, (sometimes called proposal density) denoted as π. The choice of weights compensates for the fact that samples are drawn from the importance density π rather than the posterior pdf. Any function that is positive where the posterior is positive can be used as importance density, making Dirichlet function as an ideal choice, whose implementation details are given in Sec. 5.4.1.

The objective of the work is to predict the passenger traffic in a set of Thiruvananthapuram city routes in Kerala, India using the integrated model. The methodology with which the above mathematical models are used for implementing the integrated model is explained below.

4 Methodology

The methodology of the work is shown in Fig. 3. The steps in implementing the model are

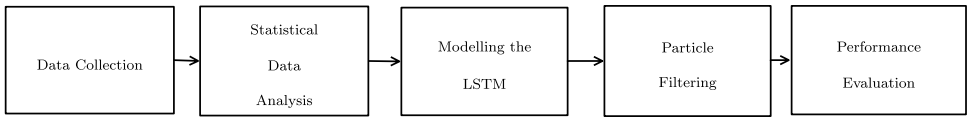


Figure 3 Methodology of work.

- Data Collection
- Statistical Data Analysis
- Modelling the LSTM network
- Particle Filtering
- Performance Evaluation

4.1 Data Collection

The ticketing data for the selected city routes are taken into consideration. The ticket data from the ticketing machines are collected, preprocessed and modelled in a form suitable for giving as an input to the Neural network. The ticket data of various routes are used to find the bus stops in each route.

4.2 Statistical Data Analysis

The time-space analysis of the data is performed to extract the features and trends in passenger traffic. The passenger count alighting and boarding from different stages in a route depends on many factors. Such time series sequential data are best modelled by RNN networks like LSTM. Three time domains are chosen for properly choosing the LSTM network *viz*, morning, noon and post noon sessions. The passenger arrival follows a poisson response, while the passenger traffic exhibits Markovian nature [1]. The Markovian nature is mathematically modelled using Bayesian Filters like Particle Filters which suits best for non gaussian nonlinear systems.

4.3 Modelling the LSTM Network and Particle Filtering

The LSTM network is the most popular and suitable RNN for performing time series data analysis problems [26]. It is chosen for predicting the passenger traffic due to following reasons

- The passenger data analysis and prediction is a sequential data analysis problem.
- The special feedback networks in the LSTM helps in processing entire sequence data in a single stretch rather than considering each single data point separately. By doing so, the network retains useful information in the data sequence to predict further sequences.
- The LSTM avoids long term dependency in the Markovian passenger dataset.

Markov processes are the basis for general stochastic simulation methods known as Markov chain Monte Carlo, which are used for simulating sampling from complex probability distributions. These methods are the basis for particle filters. So particle filters are used at the output of LSTM model, thereby it can correct the errors at the output of LSTM model.

4.4 Performance Evaluation

The performance of the model is evaluated using different criteria. The metrics used in evaluating the model are

1. Mean Absolute Error (MAE)
2. Symmetric Mean Absolute Percent Error (MAPE)
3. Coefficient of Determination,R^2
4. Kullback-Leibler (KL) divergence
5. Jensen-Shannon (JS) divergence

The absolute error is the absolute value of the difference between the forecasted value and the actual value. Mean absolute error is the mean of absolute errors. The symmetric mean absolute percentage error (SMAPE) is an accuracy measure based on percentage errors. It is percentage based and scale independent, thus that it can be used to evaluate the forecast performances of time series datasets. The lower the SMAPE value of a forecast, the higher its accuracy.

The Kullback-Leibler divergence [27] $D_{KL}(p, q)$, is the expected value of the log likelihood between p ad q. KL divergence is a measure of relative entropy between two probability distributions. It is neither a distance value nor is symmetric. Besides it does not satisfy the triangular inequality. The divergence value is calculated to prove that the statistics of the predicted value and the actual value are the same, thus validating the model.

The Jensen-Shannon divergence (JS divergence), is another method to quantify the similarity between two probability distributions. It uses KL divergence to compute a smoothed, normalised and symmetrical distant metric with scores between 0 (identical) and 1 (maximally different). The square root of the score gives the Jensen-Shannon distance.

5 Experiment

The Trivandrum city routes are selected for studying the passenger behaviour and prediction. The passenger traffic in any route is affected by many factors *viz.*, stops included *enroute*, overlapping among the routes, trip timing etc. The

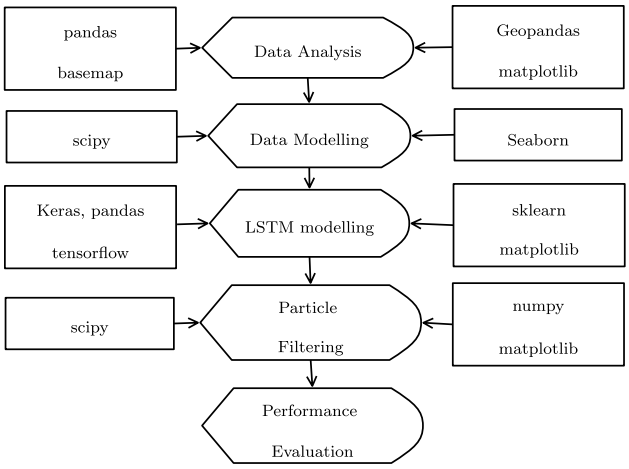


Figure 4 Python Modules used for experimentation.

experimental procedure is implemented using *Python 3.4* modules. The collected data was read using the *pandas* module, visualized to analyse the geographical extent of routes in the city. The graphical visualization is achieved using *Basemap*, *geopandas*, and *matplotlib* modules. From the ticketing data, stops of these routes were taken. The latitude and longitude values of these stops were collected from Google Map to plot them for interpreting the extent of routes. The data modelling is realized using *scipy* and *seaborn* modules. The LSTM network is modelled and trained using the *keras*, *tensorflow*, *pandas*, *sklearn* and *matplotlib* modules. The predicted output of LSTM is particle filtered using *numpy*, *scipy* and *matplotlib* modules. This setup along with the tools is shown in Fig. 4

The experimental steps in implementing the model are shown in Fig. 5

5.1 Data Modelling

The passenger data is collected from the ticketing machines. The ticket data for the period from October 2020 to August 2021 excluding May and June (lockdown period due to covid pandemic) are opted for the analysis. There are many anomalies in the ticket data like wrong entries in the trip numbering,

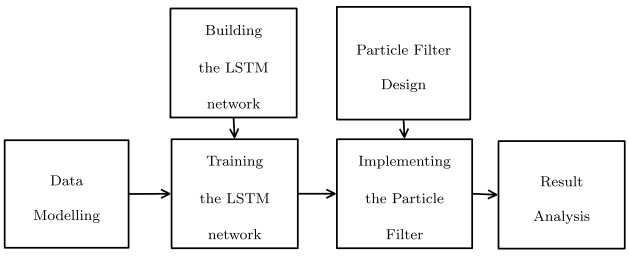


Figure 5 Steps in implementing the model.

depot numbers etc. All the files were cleaned and formatted using *pandas* and *numpy* modules. The spatial and temporal analysis are performed as shown in Fig. 6. The ticket data provides information regarding the number of stops in each route, the passengers boarding and alighting, ticket issue time and the amount collected from each passenger. The latitude and longitude data of different stops are collected for analysing the geospatial extent of routes. The timing of issuing the tickets for passengers is extracted from the ticketing data for observing the various trends and variations to perform temporal analysis. A total of 36 routes in the Thiruvananthapuram city are selected for the study. All the routes were plotted (Fig. 7a) to analyse the geographic extent and connectivity of the routes to different regions. It is clear that routes spatially extend in all the regions and connect the major parts of the city.

The Markovian behaviour of the passenger data is revealed by considering the passenger statistics in a route that connects the major city traffic from *Vizhinjam* ($8.3932°N, 77.045°E$) to *Ulloor*($8.5295°N$, $76.9290°E$). This route as shown in Fig. 7b is selected as it completely indicates the passenger travel behaviour in the city. The passenger data X_n depends only on X_{n-1} and not on the set $X_0, X_1, \cdots, X_{n-2}$. Such Markov chains are described using the transition matrix. Here the state space is defined by the likelihood of occupancy of the bus. The state space is defined as

$$S = \{S_1, S_2, S_3, S_4\} \tag{13}$$

The different states indicates the total number of seats occupied by the passengers in the bus. The state S_1 indicates the occupancy of quarterly filled seats $(0 - 25\%)$ and S_2 represents almost half filled travelling pattern $(25 - 50\%)$. The state S_3, represents the three quarter occupancy of seats $(50 - 75\%)$ in the bus by the passengers. The state S_4 is the most profitable state, that indicates almost completely filled $(75 - 100\%)$ travelling pattern. The transition matrix is calculated by analysing the traffic at various time intervals over nine months period. The obtained transition matrix is

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.8 & 0.2 \\ 0 & 0.125 & 0 & 0.875 \end{bmatrix} \tag{14}$$

The state diagram corresponding to the state transition matrix is given by The passenger occupancy in the bus initially started in the S_3 state, *i.e.*, the route started with almost filled bus, proceeded with complete occupancy of seats. As the travel proceeds and approaches the destination, the number of passengers in the bus dropped, showing a transition to state S_2 and finally S_1 (see Fig. 8).

The passenger traffic of selected routes over nine months is averaged and plotted to study the traffic model. In Fig. 9, passenger traffic from 5 : 00 till 24 : 00 hours is plotted

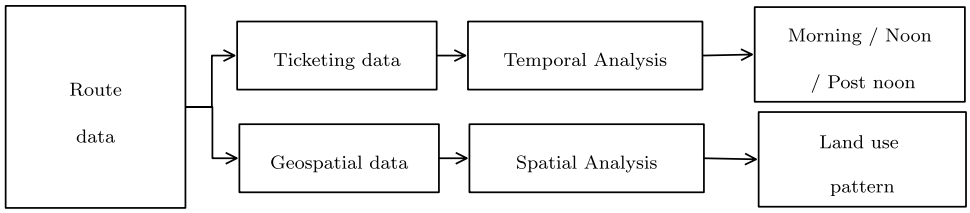


Figure 6 Steps in analysing route data.

that represents a complete single day passenger flow. It is clear from the graph that the passenger traffic shows a peak initially, then an exponential decay, again rising and then decaying gradually. Thus the data is clustered into different time based patterns. After proper analysis, three patterns are chosen, namely morning, noon and post noon session. The initial peak from 5 *am* to 10 *am* is taken as the morning session. The decaying region from 10 *am* till 1 *pm* is the noon session and from 1 *pm* till midnight is the post noon session. The histogram of these three sessions is shown in Fig. 10 which gives a clear indication that the passenger statistics is different in these three durations.

5.2 Building LSTM architecture

A deep learning based LSTM architecture is designed to extract the features of bus passenger data. In this passenger prediction approach, stacked LSTM models with three Bidirectional LSTMs (BiLSTM) and one Dense layer at the output are designed using the python deep learning library, *keras*. The first BiLSTM layer provides a 3D output as input to the subsequent BiLSTM layers. In time sequence prediction problems, BiLSTMs outperform normal LSTM layers [28] as they train the input sequences in both forward and backward directions and thus extract more features. The output of the last BiLSTM layer is given to the dense layer which builds a one-to-one relation on input and output.

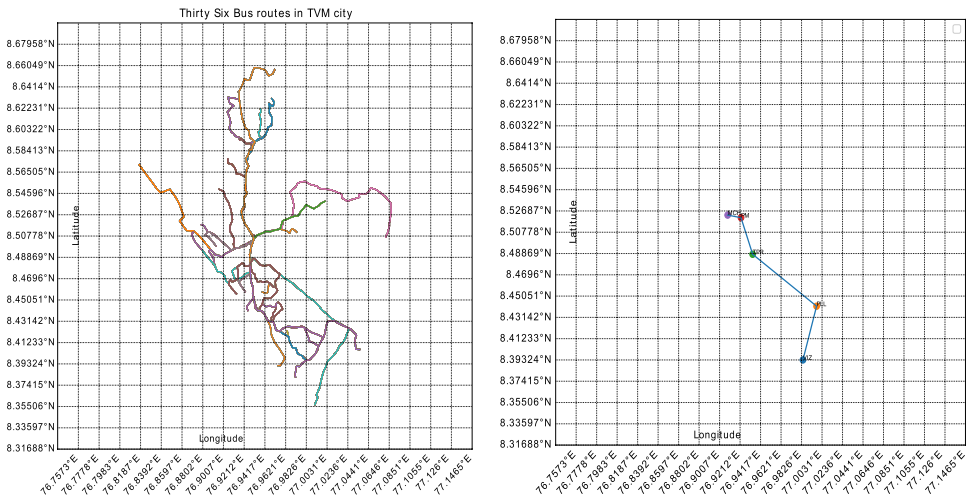
5.3 Training the LSTM Model

The modelling is performed such that the data is categorized into three, and LSTM models are trained separately for predicting the passengers in these three sessions. The whole data set is divided into training, testing and validation sets. The training set is used for training the LSTM model with sigmoid activation function. The weights and learning rate of the stacked network is updated using mean square error and RMS prop function. The RMS prop is a gradient based optimization algorithm used in recurrent neural networks. It uses the moving average of squared gradients to normalize the gradient. The vanishing gradient problem in neural networks is avoided using this normalization technique. It balances the step size (momentum) in such a way that, it decreases the step for large gradients to avoid exploding, and increases the step for small gradients to avoid vanishing. The learning rate is adaptive, it changes over time. The equations are

$$E[g^2(t)] = \beta E[g^2(t-1)] + (1-\beta)(\frac{\partial c}{\partial w})^2 \tag{15}$$

where $E[g(t)]$ is the moving average of squared gradients, $\frac{\partial c}{\partial w}$ is gradient of the cost function with respect to the weight and β is the moving average parameter. The weight updation equation is given by

$$w_{ij}(t) = w_{ij}(t-1) - \frac{\eta}{\sqrt{E[g^2(t)]}\frac{\partial c}{\partial w_{ij}}} \tag{22}$$



(a) Thiruvananthapuram city routes **(b)** Selected *Vizhinjam-Ulloor* route

Figure 7 Geographic Extent of Routes.

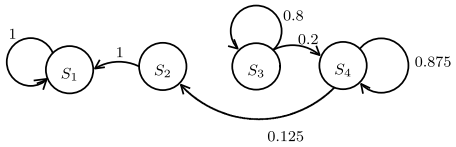


Figure 8 State transition diagram.

Here η is the learning rate. The default value of η is 0.001. The mean square error is the average of the squared differences between the actual and predicted value.

5.4 Filter Design and Implementation

The predicted data from the LSTM architecture is given as input to the particle filter to correct the errors. The particle filtering algorithm is implemented using the flowchart in Fig. 11. Initially a set of particles are generated and their weights are computed. The weights are updated using the random discrete measure to estimate the unknowns. At next time instant, before generating new particles, the effective weights of the updated particles are measured. Those with negligible measures are replaced with the ones with weights greater than effective size, *i.e.*, resampling is performed. A sequential procedure called sequential importance sampling (SIS), is used in particle filtering to obtain estimates of $P(x_{0:k}|z_{1:k})$ sequentially exploiting the Markovian nature of the state equation. The algorithm to estimate the samples of the posterior distribution by particle filtering are as follows:

5.4.1 Particle Generation and Weight Updation

The algorithm starts with the generation of particles. The samples of the probability distribution function are approximated by the combination of particles and the weights assigned to the particles. In this model, initially several thousand particles are generated, each particle representing the passenger count. Initially this is generated based on the data obtained from the predicted output from the LSTM network. The particles are represented by

$$\chi = \{x^{(m)}, w^{(m)}\}_{m=1}^{M} \tag{17}$$

where $x^{(m)}$ are the particles, more specifically the passenger count, $w^{(m)}$, their weights and M is the number of particles. Here M is chosen as 2000. Each particle is assigned a weight, that represents the probability of particles. The combination of particles and weights form a probability distribution function. The weights are assigned as

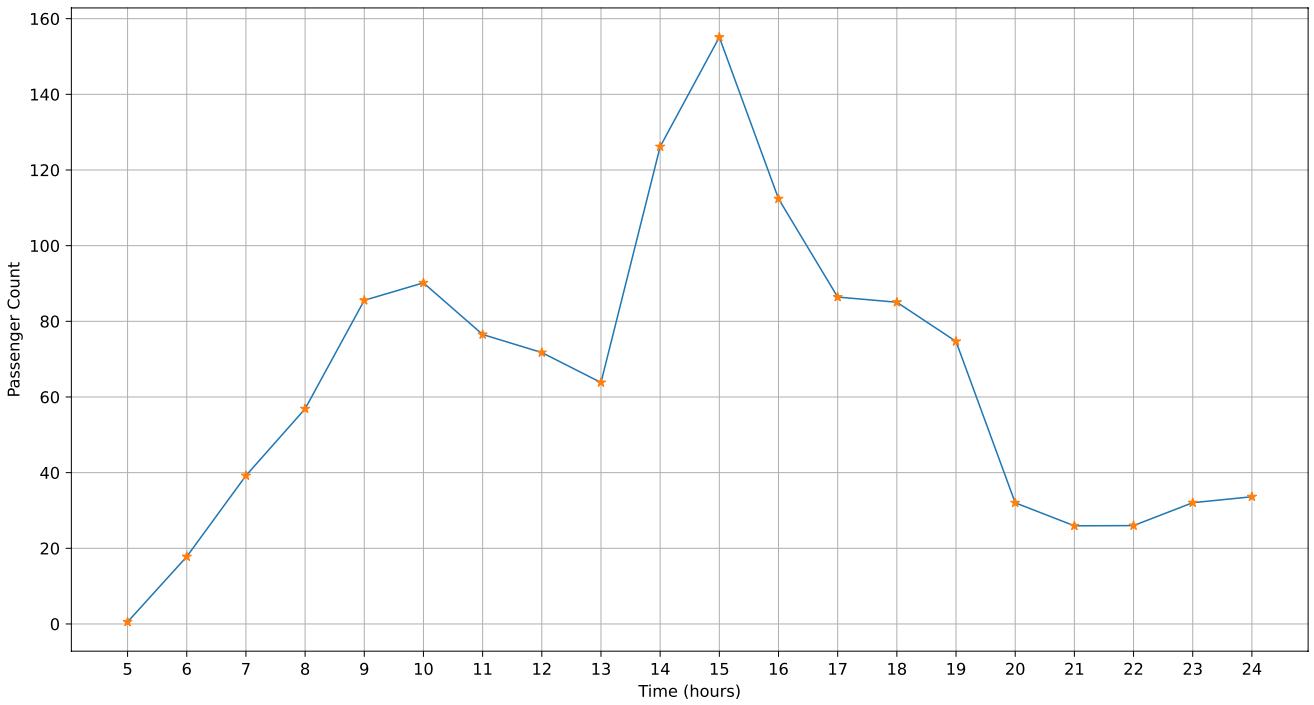


Figure 9 Single day passenger flow model.

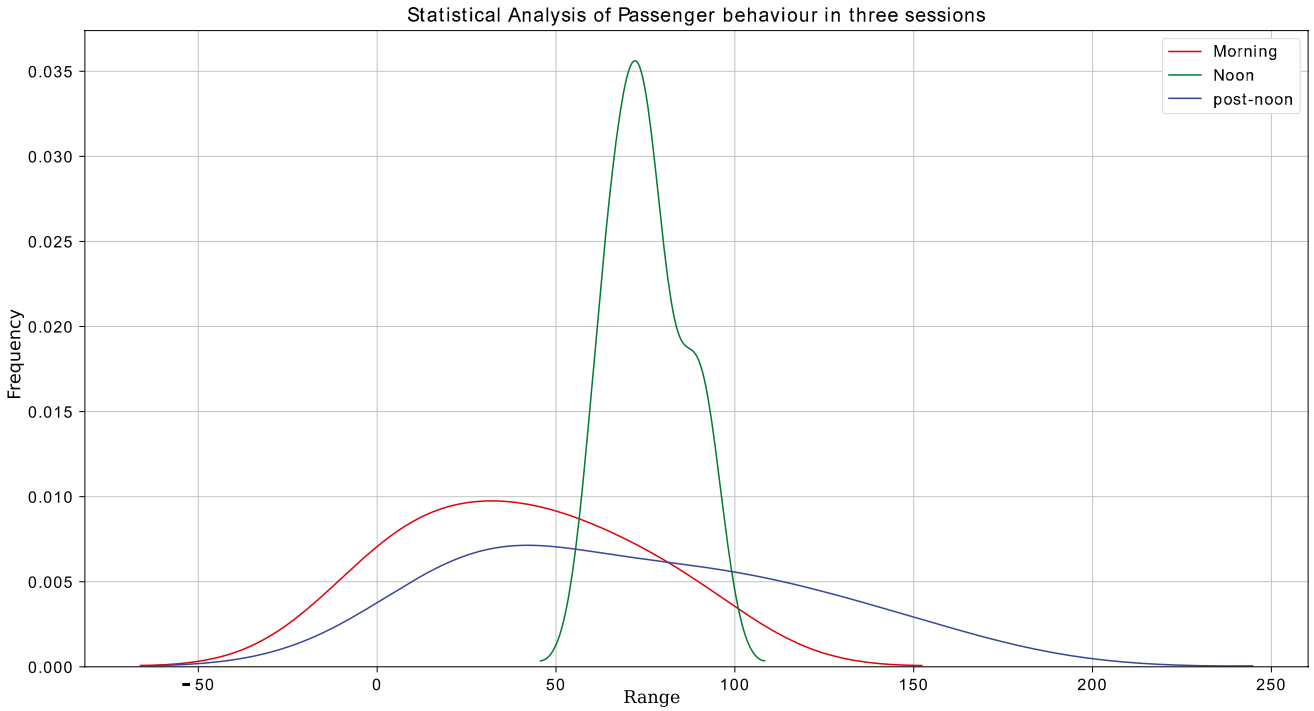


Figure 10 Histogram analysis revealing passenger behaviour in three sessions.

$$w^{*(m)} = \frac{P(x)}{\pi(x)} \tag{18}$$

The importance function $\pi(x)$ [29] can be an optimal importance function or a prior importance function, that have the same distribution as that of the probability distribution to be approximated. Using these, the probability distribution is approximated as

$$P(x) \approx \sum_{m=1}^{M} w^{(m)} \delta(x - x^{(m)}) \tag{19}$$

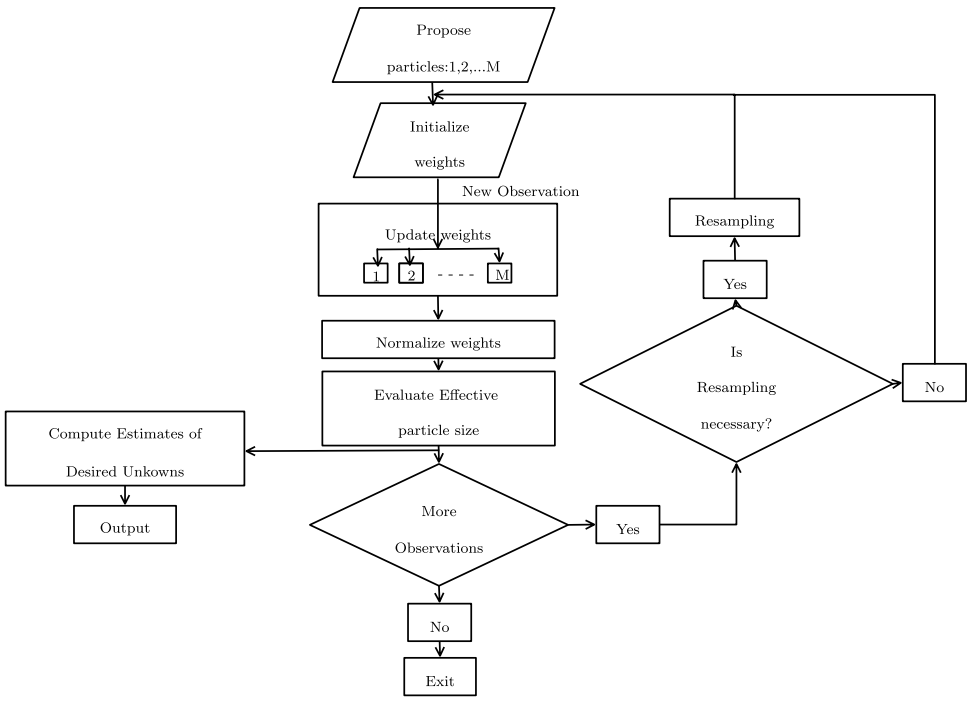


Figure 11 Particle filtering algorithm.

where $\delta(.)$ is the Dirac delta function. The function of random variable X, $g(X)$ follows the probability density function $P(x)$ and its expectation is given by

$$E[g(X)] \approx \sum_{m=1}^{M} w^{(m)} g(x^{(m)}) \tag{20}$$

The weight updation using prior importance function is given by

$$w_t^{(m)} \propto w_{k-1}^{(m)} p(y_k | x_k^{(m)}) \tag{21}$$

The weight updation using optimal importance function is as follows

$$w_k^{(m)} \propto w_{k-1}^{(m)} p(y_k | x_{k-1}^{(m)}) \tag{22}$$

where y_k and x_k represents the predicted and observed passenger data respectively at time step k. The prior is chosen over optimal importance function in the implementation of particle filters because the computation of $p(y_k|x_{k-1})$ involves following integration:

$$p(y_k|x_{k-1}) = \int p(y_k|x_k) p(x_k|x_{k-1})\, dx_k \tag{23}$$

which may not be tractable and sampling from $p(y_k|x_{k-1})$ directly may not be feasible [30].

As discussed in Sec. 3.1, the Dirichlet distribution is chosen as the prior distribution for the transition probability matrix of the passenger data that forms a Markovian chain. The Dirichlet distribution generates a random vector with length K and each element of this vector is non-negative and summation of elements is 1, meaning that it generates a random probability vector. Any function that is positive where the posterior is positive can be used as importance density, as discussed in Sec. 3.3. Thus the prior importance function is given by

$$\pi(x) \sim Dirch(p_i) \tag{24}$$

The weights of the particles thus computed are normalized, so that they sum to one, for making it into a probability distribution. The particles that are closest to the actual passenger count have a higher weight than the ones that are very much different from actual count. On normalizing, it becomes

$$w^{(m)} = \frac{w^{*(m)}}{\sum_{m=1}^{M} w^{*(i)}} \tag{25}$$

The above steps *viz.*, particle generation, prediction and updation forms the steps of Sequential Importance Sampling (SIS) algorithm. The problem associated with SIS algorithm is that it suffers from degeneracy problem. The algorithm starts with uniformly distributed particles with equal weights. As the algorithm runs, any particle that does not match the actual measurements acquires an extremely low weight. Only the particles which are closer to the passenger data have an appreciable weight. The algorithm started with 2000 particles with only few contributing meaningfully to the state estimate *i.e.*, the filter has degenerated. This degeneracy, that deteriorates the performance of particle filters can be reduced by resampling.

5.4.2 Particle Resampling

The resampling algorithm operates in such a way that, it discards particles with very low probability and replaces them with new particles with higher probability. It does that by replicating particles with relatively high weights. These replicated ones are slightly dispersed by the noise added in the predict step, which results in a set of points in which a large majority of the particles accurately represent the probability distribution as shown in Fig. 12.

There are many resampling algorithms. Standard resampling algorithms are simple random resampling, systematic resampling, residual resampling and branching corrections. Here simple random resampling, also called multinomial resampling is used. It samples from the current particle set N' times, making a new set of particles from the sample. The probability of selecting any given particle is proportional to its weight. This is implemented using *NumPy's cumsum* function. It computes the cumulative sum of an array.

The resampling is not done at every epoch. It is performed after calculating the effective M, which measures the number of particles that effectively contribute to the probability distribution. The equation for this is

$$M_{eff} = \frac{1}{\sum_{m=1}^{M} (w^m)^2} \tag{26}$$

If M_{eff} falls below the threshold value, it is time to resample.

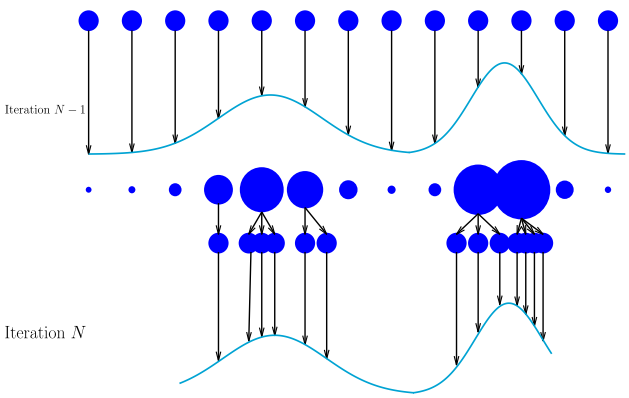


Figure 12 Resampling algorithm.

Thus the resampling procedure is summarized as follows

- Draw M particles $x_k^{*(m)}$ from the discrete distribution $p(x)$ and assign equal weights $(\frac{1}{M})$ to the particles.
- The new weights are updated using importance sampling algorithm based on Eq. 21
- After weight updation, particles with negligible weights are replaced by replicas of particles with larger weights as shown in Fig. 12. In the figure, sizes of the particles reflect the weights assigned to them.

5.5 Performance Analysis

The result is analysed using the following performance metrics. The actual and predicted values are analysed using Eqs. 27a to 27c. Here y_k is the observed passenger flow and x_k is the predicted flow during k^{th} time interval. N is the total number of predicted passenger data. The statistics of the predicted values are analysed using Eqs. 27d and 27e. Here P and Q denotes the probability distribution of actual and predicted passenger count.

$$MAE = \frac{1}{N}\sum_{k=1}^{N}|y_k - x_k| \tag{27a}$$

$$SMAPE = \frac{2}{N}\sum_{i=1}^{N}\frac{|y_k - x_k|}{y_k + x_k} \times 100\% \tag{27b}$$

$$\text{Coefficient of Determination, } R^2 = 1 - \frac{\sum(y_k - x_k)^2}{\sum(y_k - \bar{y})^2} \tag{27c}$$

$$D_{KL}(P||Q) = \sum_{x \in X} \log\left(\frac{P(x)}{Q(x)}\right) \tag{27d}$$

$$\text{JSD}(P||Q) = \frac{1}{2}D(P||M) + \frac{1}{2}D(Q||M) \tag{27e}$$

where $M = \frac{1}{2}(P + Q)$

6 Results

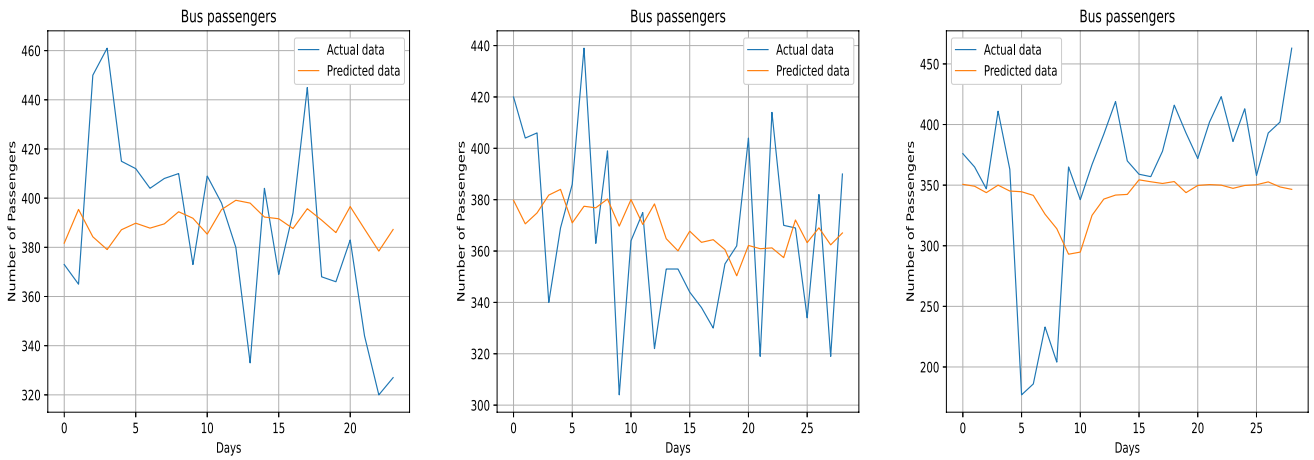
The LSTM model is used initially to predict the passenger data during morning,noon and post noon sessions. The dataset for the three sessions are used separately to train the LSTM model. The outputs are shown in Fig. 13

The training data used to train the LSTM model is very small, thus the result indicates great deviation from the actual values. The training and validation loss with respect to time is plotted (Fig. 14). In the morning session (Fig. 14a), the validation loss is much better than the training one, indicating that the validation data is scarce. In the noon session (Fig. 14b), training loss and validation loss are little bit high, showing poor model performance. In the post noon session (Fig. 14c), curve moves noisily, not showing a steady performance.

As the dataset is very small, the performance of the LSTM model is poor, which is visible in the output graphs. This is rectified using an integrated model which combines an LSTM and a particle filter. The output of the LSTM network is filtered by the particle filter algorithm and the results are improved (Fig. 15). The output accuracy, indicated by the R^2 value of the integrated model predicting the passengers travelling during the morning time (Fig. 15a) is 0.857, indicating a promising performance.

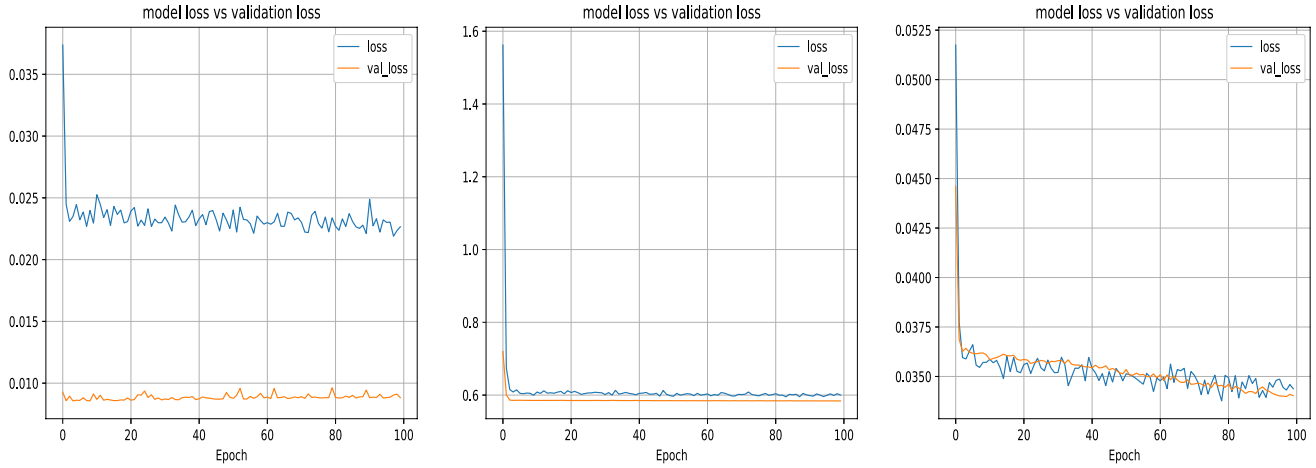
The accuracy for the predicting the passenger count during noon session (Fig. 15b) is 0.784 and that during the post noon (Fig. 15c) is 0.876.

These results when compared with the output of LSTM alone (Fig. 13) shows remarkable difference in the output,



(a) Morning session **(b)** Noon session **(c)** Postnoon session

Figure 13 Passenger data prediction using LSTM alone.



(a) Morning session **(b)** Noon session **(c)** Postnoon session

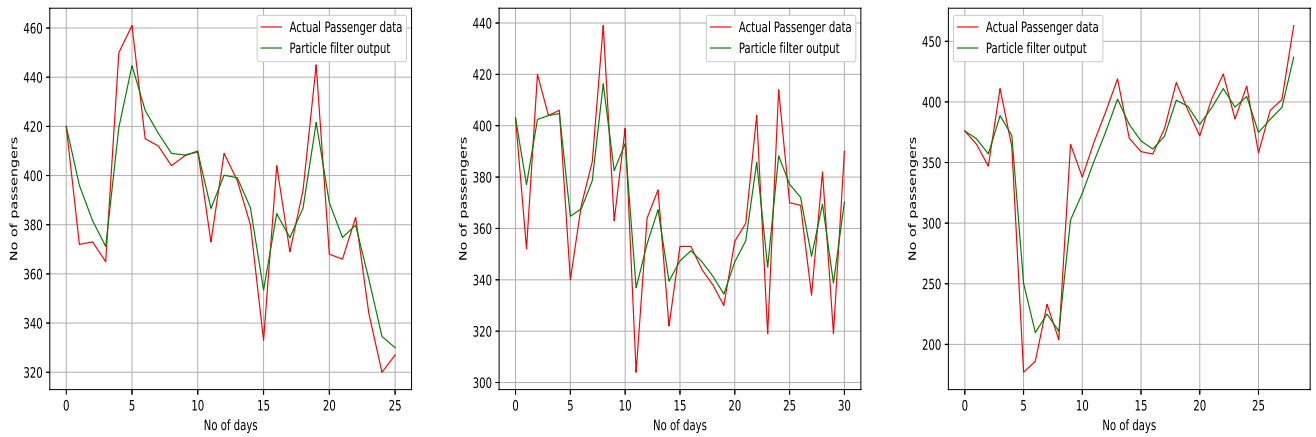
Figure 14 Training and validation loss curve.

which is validated using different parameters and the results are presented in Table 1.

In Table 1, it is observed that the Mean Absolute error considerably reduced to 3.81%, 3.93% and 6.38% for the integrated model considering the temporal sessions separately. Here the absolute error is considered as the size of the dataset is small, and the error can be indicated relative to the size of the passenger dataset. The SMAPE of the integrated model dropped to 9.4%, 7.8% and 4.6% for the three sessions indicating the effectiveness of the model. A good R^2 value of 85%, 79% and 89%, is an indication that the chosen model correctly follows the trends in the passenger data and is predicting well.

The histogram of the actual and predicted values are initially plotted to model the probability mass function for calculating the divergence values. The histogram is then normalised to obtain the probability mass function (pmf) of both the predicted and the actual values. The pmf of the signals in the morning session is shown in Fig. 16.

These pmf values are used to compute the KL divergence values. The KL value for the morning session is obtained as 1.70, indicating a better match between the predicted and actual distribution. While for the LSTM model alone, KL value obtained is 2.94 which is higher than that for the integrated model. The pmf of the signals in the noon session is shown in Fig. 17.



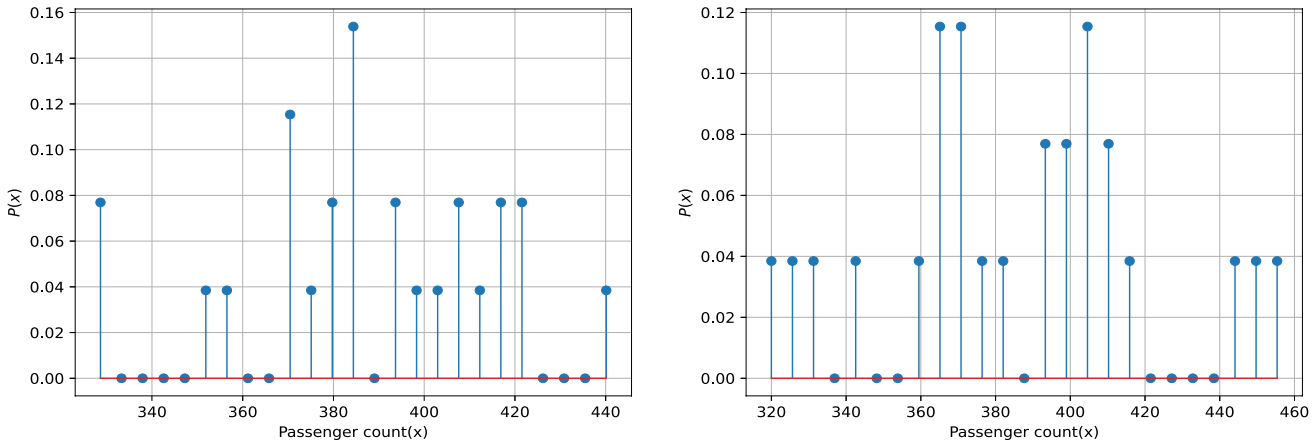
(a) Morning session **(b)** Noon session **(c)** Postnoon session

Figure 15 Predicted passenger count of the integrated model.

Table 1 Evaluation of model performance.

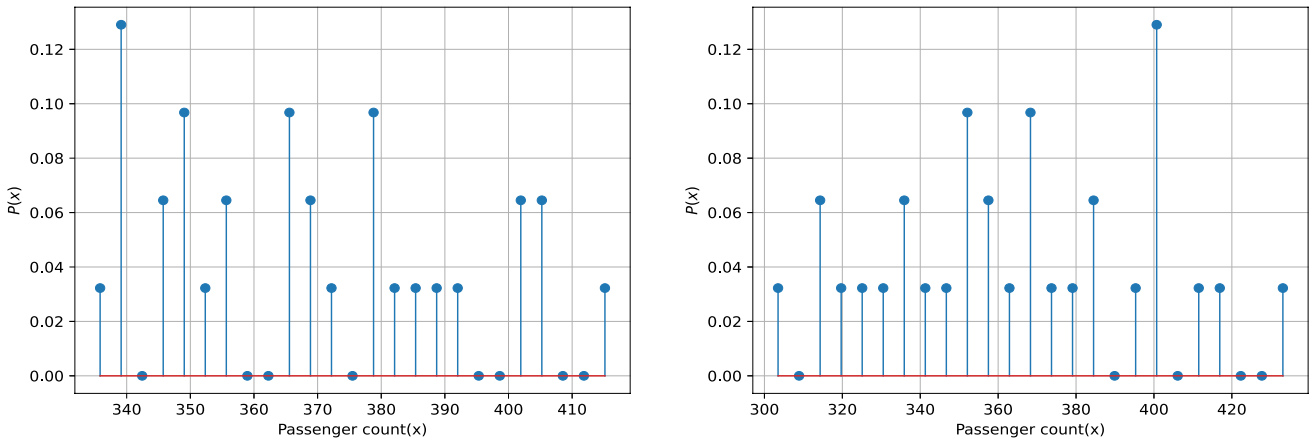
Parameters	Morning		Noon		Postnoon	
	LSTM	LSTM +PF[a]	LSTM	LSTM +PF	LSTM	LSTM +PF
MAE	10.33%	3.81%	46.50%	3.93%	29.99%	6.38%
SMAPE	17.90%	9.40%	18.90%	7.80%	16.60%	4.60%
R^2	0.04	0.85	0.05	0.79	0.04	0.87
KL Divergence	2.94	1.70	2.61	1.78	2.94	1.57
JS Divergence	0.31	0.19	0.22	0.18	0.29	0.17

[a]Particle Filter



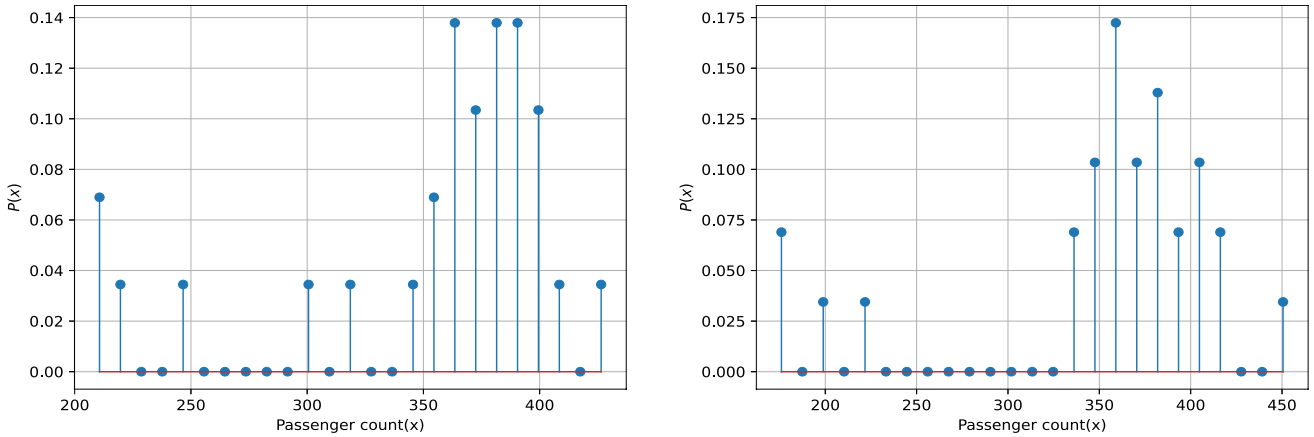
(a) Predicted signal **(b)** Actual signal

Figure 16 Probability mass function during morning session.



(a) Predicted signal **(b)** Actual signal

Figure 17 Probability mass function during noon session.



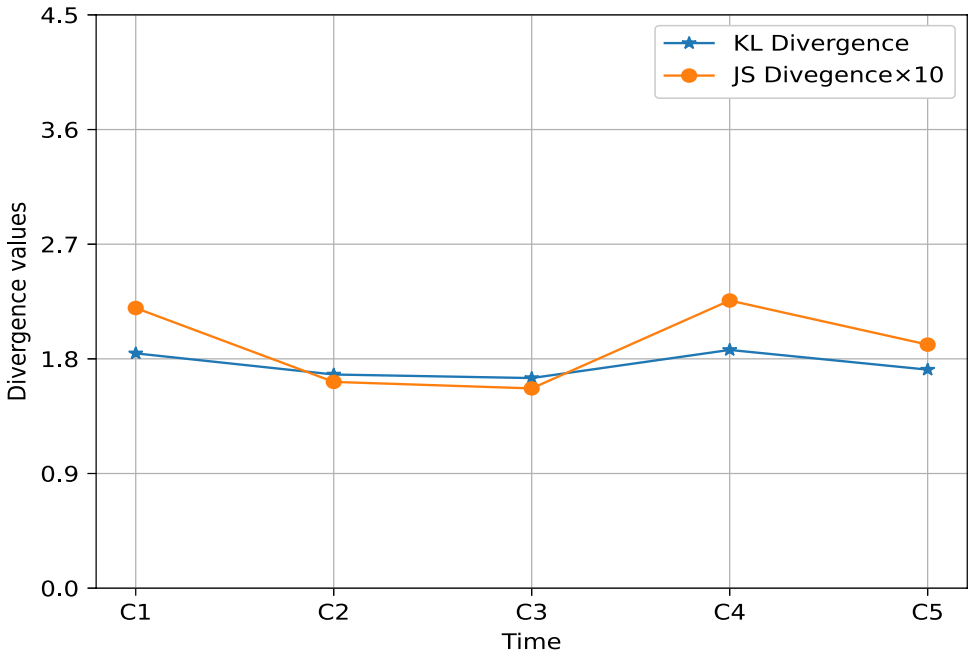
(a) Predicted signal **(b)** Actual signal

Figure 18 Probability mass function during post noon session.

The KL value computed using these pmfs is 1.78, thus the predicted distribution almost follows actual distribution. For the LSTM model, value is 2.61, proves that the integrated model is better. The pmf of the signals in the post noon session is shown in Fig. 18 with KL value of 1.57, which is almost half of that obtained for LSTM model alone.

The Jenson-Shannon Divergence is an extension of KL divergence, which is a symmetrical score and distance measure of one pmf over the other. It is the normalized score of KL divergence, becomes symmetrical. The square root of the JS Divergence gives Jenson-Shannon distance, or JS distance. The JS distance obtained for three sessions are 0.43, 0.42 and 0.41 respectively indicating the prediction accuracy of the integrated model. To validate the model performance, the temporal clustering of the passenger data is remodelled and five sessions were chosen. The passengers travelling from morning 4 *am* to 8 *am* is chosen as C1, those from 8 *am* to 12 noon is C2, those travelling in the afternoon session from 12 noon to 4 *pm* is C3, those who are plying the buses in the evening time from 4 *pm* to 8 *pm* are C4 and C5 are the one who chose to travel at night from 8 *pm* to 12 midnight. The KL and JS divergence values for the predicted and actual passenger probability density functions have been computed and plotted as shown in Fig. 19. The linearity in the divergence

Figure 19 Divergence values.



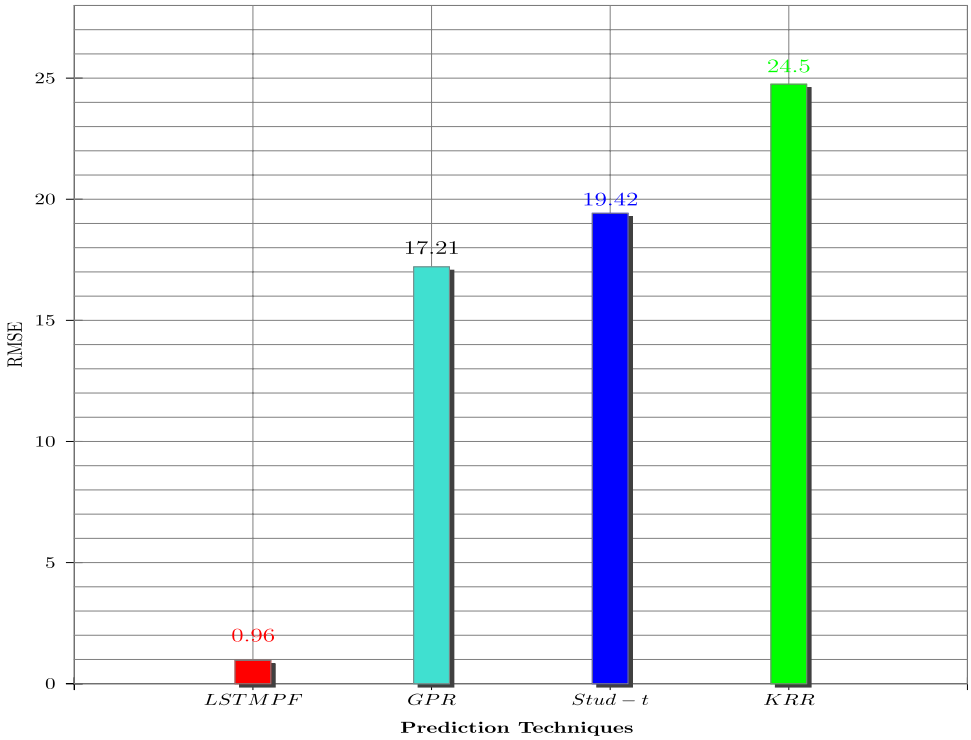


Figure 20 Performance Comparison.

values at different time intervals, is an indication that the model is performing well in predicting the passenger count.

The performance of the proposed LSTM-PF model is evaluated with the existing prediction models like Gaussian Process Regression (GPR) model, Student-t regression model and Kernel Ridge Regression (KRR) model based on the root mean square error (RMSE) value of the predicted values. From Fig. 20, it is clear that the proposed hybrid model is highly effective in predicting the passenger traffic compared to other models.

7 Conclusion and Future Scope

This paper proposes an integrated model using LSTM and particle filter for predicting the bus passenger traffic. The passenger data flow is complex and random in nature, influenced by many fixed and stochastic parameters. The passenger traffic exhibits Markovian behaviour, as visible in the experimental results. The state transition diagrams obtained from the Markovian analysis reveals the required states to be maintained while running a route for a better epk (earning per kilometer). These results necessitate the implementation of circular routes within the city so that the bus is almost completely occupied during the whole travel time. The models based on three temporal patterns resulted in better prediction with LSTM coupled with particle filters. This work is a step towards facilitating automatic bus scheduling, based on passenger demand, that will ultimately lead to an Intelligent Transport Systems.

Author Contributions All authors have made substantial contributions to the conception and design of the work; VGS performed the acquisition, analysis, and interpretation of data. VGS and HVS have done the creation of new software used in the work; VGS have drafted the work and HVS substantively revised it. All authors read and approved the final manuscript.

Funding The authors did not receive support from any organization for the submitted work.

Availability of Data and Material The data that support the findings of this study are available from Kerala Road Transport Corporation (KSRTC) but restrictions apply to the availability of these data, which were used under licence for the current study, and so are not publicly available. Data are however available from the authors upon reasonable request and with permission of KSRTC.

Declarations

Conflicts of Interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

1. Vidya G. S., & Hari, S. (2022). *Journal of Signal Processing Systems*. https://doi.org/10.1007/s11265-022-01774-3
2. Nagaraj, N., Gururaj, H., Swathi, B., & Hu, Y. C. (2022). *Multimedia Tools and Applications*, *81*. https://doi.org/10.1007/s11042-022-12306-3
3. Liu, Y., Cheng, J., Zhang, H., Zou, H., & Xiong, N. (2020). *IEEE Access, 8*, 216245. https://doi.org/10.1109/ACCESS.2020.3041294

4. Ahmed, M. S., & Cook, A. R. (1979). *Transportation Research Record*.
5. Karlaftis, M., & Vlahogianni, E. (2011). *Transportation Research Part C: Emerging Technologies, 19*, 387. https://doi.org/10.1016/j.trc.2010.10.004
6. Ge, S., Zheng, C., & Hou, M. (2013). *Applied Mechanics and Materials, 433–435*, 1374. https://doi.org/10.4028/www.scientific.net/AMM.433-435.1374
7. Singh, A., Obaidat, M. S., Singh, S., Aggarwal, A., Kaur, K., Sadoun, B., Kumar, M., & Hsiao, K. F. (2022). *Simulation Modelling Practice and Theory*, *121*, 102658. https://doi.org/10.1016/j.simpat.2022.102658. https://www.sciencedirect.com/science/article/pii/S1569190X22001289
8. Baudoin, F. (2010). In P. Peterson, E. Baker, & B. McGaw (Eds.), *International Encyclopedia of Education (Third Edition)* (3rd ed., pp. 451–452). Elsevier, Oxford. https://doi.org/10.1016/B978-0-08-044894-7.01369-5. https://www.sciencedirect.com/science/article/pii/B9780080448947013695
9. Elfeki, A., & Dekking, M. (2001). *Mathematical geology, 33*(5), 569.
10. Fitzgerald, W. J. (2001). *Signal Processing, 81*(1), 3.
11. Myers, D. S., Wallin, L., & Wikström, P. (2017). MVE220 Financial Risk: Reading Project.
12. Wu, Y., Wang, D. Z., & Zhu, F. (2022). *Physica A: Statistical Mechanics and its Applications*, *593*, 126989. https://doi.org/10.1016/j.physa.2022.126989. https://www.sciencedirect.com/science/article/pii/S0378437122000772
13. Liu, S., Zhou, H., & Cheng, X. (2019). *Mobile Networks and Applications, 25*, 1. https://doi.org/10.1007/s11036-019-01405-5
14. Raheja, S., Obaidat, M. S., Kumar, M., Sadoun, B., & Bhushan, S. (2022). *Simulation Modelling Practice and Theory*, *118*, 102540. https://doi.org/10.1016/j.simpat.2022.102540. https://www.sciencedirect.com/science/article/pii/S1569190X22000417
15. Liu, S., Chen, X., Li, Y., & Cheng, X. (2019). *Symmetry, 11*, 1471. https://doi.org/10.3390/sym11121471
16. Toqu, F., Khouadjia, M., Come, E., Trepanier, M., & Oukhellou, L. (2017). In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)* (pp. 560–566). https://doi.org/10.1109/ITSC.2017.8317939
17. Chhabra, M., Ravulakollu, K., Kumar, M., Sharma, A., & Nayyar, A. (2022). *Neural Computing and Applications*, 1–27. https://doi.org/10.1007/s00521-022-07894-y
18. Ch, P., Stephan, T., Kumar, M., & Nayyar, A. (2022). *Neural Computing and Applications*, *34*. https://doi.org/10.1007/s00521-022-07515-8
19. Huang, Y. C., & Ma, M. Y. (2022). *Measurement and Control, 55*(9–10), 881. https://doi.org/10.1177/00202940221083574.
20. Hua, Y., Zhao, Z., Li, R., Chen, X., Liu, Z., & Zhang, H. (2019). *IEEE Communications Magazine, 57*(6), 114. https://doi.org/10.1109/MCOM.2019.1800155
21. Jiao, F., Huang, L., Song, R., & Huang, H. (2021). *Sensors*, *21*(17). https://doi.org/10.3390/s21175950. https://www.mdpi.com/1424-8220/21/17/5950
22. Wang, J., & Wen, C. (2022). *Sensors*, *22*(7). https://doi.org/10.3390/s22072574. https://www.mdpi.com/1424-8220/22/7/2574
23. Wang, P., Chen, X., Chen, J., Hua, M., & Pu, Z. (2021). *IET Intelligent Transport Systems*, *15*(2), 248. https://doi.org/10.1049/itr2.12018. https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/itr2.12018
24. Qin, Y., Li, Y., & Liu, G. (2022). *Remote Sensing*, *14*(18). https://doi.org/10.3390/rs14184629. https://www.mdpi.com/2072-4292/14/18/4629
25. Manaswi, N. K. (2018). In *Deep Learning with Applications Using Python* (pp. 115–126). Springer.
26. Hochreiter, S., & Schmidhuber, J. (1997). *Neural Computation, 9*(8), 1735. https://doi.org/10.1162/neco.1997.9.8.1735
27. Chou, R., Boers, Y., Podt, M., & Geist, M. (2011). Fusion 2011 - 14th International Conference on Information Fusion.
28. Abduljabbar, R. L., Dia, H., & Tsai, P. W. (2021). *Journal of Advanced Transportation*, *2021*.
29. Doucet, A., de Freitas, N., & Gordon, N. J. (2001). In *Sequential Monte Carlo Methods in Practice*.
30. Theodoridis, S. (2020). In S. Theodoridis (Ed.), *Machine Learning (Second Edition)* (2nd ed., pp. 871–900). Academic Press. https://doi.org/10.1016/B978-0-12-818803-3.00029-5. https://www.sciencedirect.com/science/article/pii/B9780128188033000295

Vidya G S is a research scholar with Department of Electronics, College of Engineering Chengannur. Her research interests are in data analysis for intelligent transport systems and smart cities.

Dr. Hari V S is Associate Professor with Department of Electronics, College of Engineering Chengannur. His research interests are in data analysis, machine learning and computer vision.