Intelligent Traffic Flow Prediction Using Optimized GRU Model

Introduction

Intelligent Transportation System (ITS) is an essential element of smart cities. ITS not only provides real time traffic characteristics but also predicts short-term traffic flow. Traffic prediction is useful for route diversions and congestion control. However, due to the stochastic nature of transportation data, time-series prediction based on huge and real-time data provided by road sensors is a challenging task. These challenges are noticed in data representation, model validation, optimal prediction framework building, estimation time-lag ahead, and the effect of external factors on traffic models. Various techniques are proposed by scientists to validate traffic predictions. Classic mathematical and statistical approaches mainly apply linear models and shallow machine learning techniques to estimate traffic characteristics. Those are not accurate because of unpredictable traffic flow variations which are essentially nonlinear and random in nature. It is trivial to anticipate the usual traffic congestion ahead of time. However, it requires significant effort to predict the traffic flow to a given degree of accuracy in a given time interval. Such statistical models use the structured expressions which are predetermined and make assumptions about the variables. These assumptions sometimes turn out to be correct resulting in correct prediction, and sometimes perform badly. The data-driven deep neural models are popular to predict traffic flows. The paper aims to fine-tune the existing GRU based deep neural networks to increase their accuracy of traffic flow predictions. These tuned sets of configuration values for the neural network are called hyperparameters. In this paper, these hyperparameters for GRU deep neural network are searched and the values of these hyperparameters are found to improve its prediction accuracy. The proposed method improves hyperparameters and sliding window steps and further evaluated the prediction accuracy and stability. The main contributions of this paper are described below:

1) A search mechanism for recurrent neural network is proposed that results in obtaining network hyperparameters, optimized number of sliding window steps ahead and improve the prediction accuracy by reducing the error in traffic flow prediction.

2) Once refined time-lags and hyper parameter set is obtained by applying the proposed mechanism, this study further evaluates and compares the accuracy of GRU with various deep learning techniques, for example, bidirectional, stacked and convolutional variants of LSTM for traffic prediction.

3) Experimental Result applied on a public traffic dataset shows an improved accuracy with the proposed method by applying a search mechanism to fine-tune the neural network models.

Methodology

A. Time-series prediction problem

Given an input sequence $X = \{x_1, x_2, \ldots, x_k\}$ as original data, where $x_i \in R^d$. We choose a sliding window of length L to define characteristics sequences from original sequence X such that $X = (X_1, X_2, \ldots, X_B)$ where $X_i = (x_i, x_{i+1}, \ldots, x_{i+L})$, where each $X_p \in X$, R^L . The historical or ground values are given by $y = (y_1, y_2, \ldots, y_{k-1})$ where $y_p \in R^1$. Alternatively, we can denote it by $y = \{y_i : i \in T\}$ where T is called the index set. The goal here is to predict the next value denoted by $y^{T\hat{}}$. A nonlinear function f is learned by mapping the temporal sequence feature X and its corresponding ground-truth value y to obtain the estimated value $y^{T\hat{}}$ with the following formulation:

$$y^{T\hat{}} = f(X, y) \qquad (1)$$

where f is the nonlinear mapping function and learning f our goal using GRU Neural Network model in this work.

B. The framework

Firstly, traffic flow data from PeMS database is drawn. PeMS contains Internet Of Things (IoT) based time sequences of traffic data recorded by the road sensors installed across the highways. Dataset is processed to obtain univariate traffic flow sequences. Original data is real-time observations of average traffic flow obtained every 5-minutes interval. Data is further cleaned up by obtaining hour level details for simplification and the idea is to reduce the processing costs. Training is then preformed using proposed algorithm for GRU optimization, Network tuning is performed in order to obtain better set in each iteration and after getting better hyperparameters. Prediction is performed using defined set of performance metrics to compare the accuracy.

C. The algorithm

Start by data sequence denoted by X = Fd07−20 which represents traffic flow time-series data obtained from PeMS website ranges from 07thJan − 20thJan, 2019 for freeway 9, district 10, California, USA. L denotes the number of steps of sliding windows, η is network learning rate, Xt and Xv denotes the training and validation sets over X. Traffic flow for Training and Validation sets based Optimization algorithm requires lower error rate. An algorithm is proposed to fulfill dual objective, that is, tune the hyperparameters (learning rate) and search for the sliding window time-step for better validation and prediction. The input to the algorithm is an original traffic flow time-series sequences; a set of sliding window of length L, Learning Rate η from a finite set. Various seed values for # of neurons of input layer, activation function, batch window size, # of epochs, optimizer function and loss functions for GRU network is initialized. Our goal is to obtain an optimized set of neural network parameters and hyperparameters.

D. The parameter settings

There are various neural network parameters and hyperparameters. Finite set of values for each parameter for basic GRU model is defined. Important network parameters are: the number of time-lag window steps L and the size of input neurons and the batch-size b in training process. The size of hidden neurons for each layers in GRU are kept fix to 64 and 32 respectively in this search experiment.

Experimental results

After data is picked up from real-time traffic flow time series dataset, python environment is used with Anaconda installed on the Windows 10, 64 bit machine. Experiment required scikit-learn, Pandas, NumPy and Matplotlib extension libraries installed.

A. Dataset description

Experimental data is obtained from Caltrans Performance Measurements Systems (PeMS) dataset, which offers historical database of traffic flow in California. The data describes the traffic flow of different car lanes of San Francisco bay area freeways. There are more than 39000 individual sensor detectors spanning the freeway system across all major metropolitan areas of the State of California and its dataset is widely used for researcher to develop and evaluate traffic models. The collected traffic data is a 5-min interval for traffic flow measures of District 10 in the first whole month of 2019. The collected traffic flow data is divided into two parts, i.e., training data and test data. The test data is used to predict traffic flow and evaluate the model's accuracy.

B. Processing of traffic series

Traffic flow data as time-series captured for 13 consecutive days, the task is to predict the traffic flow representing number of vehicles after every 5 minutes slot using GRU and other neural networks after obtaining tuned hyperparameters. The traffic flow data considered for the District 10 and Freeway 99 and as mentioned ranges precisely from January 7, 2019 to January 20, 2019 for experiment purpose. The dataset is split into training, validation and testing subsets with the ratio of 80/20/20 respectively in the work.

C. Algorithm execution

The dataset can be loaded using Python Library. Once data is ready for further processing, various settings to optimize the search for parameters and hyperparameters of the network is defined. Important network parameters are: the number of sliding window steps L and the size of input neurons, hidden units for each layers in GRU are set to 64 and 32 respectively. To demonstrate the better performance of the model, a grid like search mechanism was conducted defined in the algorithm over Number of neurons in first layer $N \in \{64, 128, 256, 512, 1024\}$, sliding-window steps $L \in \{3, 6, 12, 18, 24\}$, and learning rate $\eta \in \{0.01, 0.05, 0.1, 0.2, 0.5, 0.8, 1.0\}$.

D. Model configurations

Python computation library is used for numerical analysis, to build GRU neutral network models and evaluate the single-step time-series traffic flow prediction performance. Further Talos python library is used to preform initial hyperparameters tuning for GRU model optimization along with our proposed algorithm for tuning window time-steps. The GRU network is built by four dense layers. Second, third and fourth layers are kept fixed with 64, 32 and 1 neurons respectively. Proposed algorithm is applied to obtain a superior set of learning rate, sliding window length, number of neurons and optimization technique. Five experiments were conducted such that starting with GRU training & validation optimization framework for PeMS traffic data ranges from 07-Jan-2019 to 20-Jan-2019 at Freeway 99, District10, California. In the first step, an open search was performed keeping parameters bounded to subset values to save processing time. GRU configurations used in these experiments are: activation: RELU, loss: MSE, optimizer: Adam, Last Layer Activation: Sigmoid and Batch Size: 64 and 256 respectively. The following set with better performance was observed $\{\eta = 0.1$, BatchSize = 64, Slidingwindowsteps = 6, Epochs = 100,Neurons = 256$\}$. Next experiment was conducting with already searched best settings found in previous step. The number of learning rate in each row is set as the η. Now one parameter variable is deliberately kept, the others are fixed - in this case the learning rate $\eta \in \{0.01, 0.05, 0.1, 0.2, 0.5, 0.8, 1.0\}$. The experiment results in obtaining $\eta = 2$ performed better than others. Next experiment is performed with already searched best settings found till previous step. L is set as the number of sliding window steps ahead used in each row. One parameter is kept as variable, the others are kept fixed- in this case the window steps $L \in \{3, 6, 12, 18, 24\}$. The experiment results in obtaining $L = 12$ performed better than others on proposed dataset. Eventually, last experiment is performed with already searched best settings by evaluating the two optimizer functions 'Adam' and 'Nadam'. The optimizer 'Adam' performed better than 'Nadam'.

E. Computational analysis

Both LSTM and GRU are widely known variants of RNN. They are used for traffic flow prediction using temporal dependency. This process uses time-lags or sliding window that essentially feed the data to neural network step by step. Having used the various deep learning variants of LSTM, to compare accuracy and computation time during training by using the fine-tuned hyperparameters for vanilla LSTM, Stacked LSTM, bidirectional LSTM and GRU models. These models are used to process the time series traffic data and predict the traffic flow. It can be observed that GRU possess training time lesser that other models. This is because GRU by virtue of

design is a simpler model with lesser number of gates. Bidirectional LSTM model performed badly with traffic time-series based data due to the reason of processing data in two directions.

F. Evaluation metrics

To evaluate the effectiveness of the proposed GRU method, three performance metrics are used: the mean absolute error (MAE), the root mean square error (RMSE) and the mean absolute percentage error (MAPE). MAPE is the sum of the individual absolute errors divided by the demand. It is the average of the percentage errors. The Mean Absolute Error (MAE) is a popular KPI to measure forecast accuracy and is the mean of the absolute error. The Root Mean Squared Error (RMSE) is another KPI, very helpful in time-series predictions and is defined as the square root of the average squared error. Below is the mathematical expression of these measures. $e_T = y^{T\hat{}} - y^T$ in below expressions is the error defined by the difference between predicted value and ground truth. The mathematical expression is the three metrics RMSE, MAE and MAPE are represented by following equations:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_T^2} = \sqrt{\frac{1}{n} * \sum_{t=1}^{n} (\hat{y^T} - y^T)^2} \quad (2)$$

$$MAE = \frac{1}{n}\sum_{t=1}^{n} |e_T| = \frac{1}{n} * \sum_{t=1}^{n} |\hat{y^T} - y^T| \quad (3)$$

$$MAPE\% = \frac{100}{n}\sum_{t=1}^{n} \left|\frac{e_T}{y^T}\right| = \frac{100}{n} * \sum_{t=1}^{n} \left|\frac{\hat{y^T} - y^T}{y^T}\right| \quad (4)$$

AVG1 represents the average value of RMSE, MAPE and MAE measures for the experiment conducted to determine traffic prediction with common or untuned parameter settings. Similarly, AVG2 represents the average value of RMSE, MAPE and MAE for the experiment traffic prediction with optimized parameter settings. The Gain in average error is presented by following expression:

$$Gain = |AVG2 - AVG1| \quad (5)$$

The error e_T can be a positive or negative depending upon the forecast overshoots or undershoots the ground truth, however, in these measures we are more interested in the error magnitude. To evaluate the performance of Proposed model, four effectiveness metrics are used. Various traffic prediction research articles use unanimous set of evaluation metrics because it is an intricate task. The most common metrics are RMSE, MAPE and MAE. A problem with them is that they fail to provide comparable measures when data is multi-dimensional, and complexities of the network models taken in experiment are divergent. Another aspect is about spatio-temporal nature of the traffic data, when trained with the neural network models, errors can possibly propagate through time and space dimensions during training. Therefore, proposing a measure like average error gain is a motivation to help measuring model's performance by describing a diversified correlation.

G. Prediction results and comparison of models

To obtain better parameters from the sample space, successive experiments on the GRU model were performed. The search was performed using hourly samples to reduce the processing cost instead of original 5-minutes observations found at PeMS traffic data. Assuming that the parameters will perform well for other DL models which use similar sliding window technique for prediction, this approach was tested with six models: GRU, Vanilla LSTM, Stacked LSTM, Bidirectional LSTM, CNN-LSTM and ARIMA. The first experiment is conducted as follows: The performance comparison with different prediction models before tuning of hyperparameters is performed by choosing a random setting from the sample space such that: No. of Neurons in first

layer 64, window steps 2, Epochs 1000, Batch size 128 and Learning rate 0.01. For efficient measurement three metrics were evaluated: RMSE, MAPE, and MAE. The second experiment is performed using the optimized hyperparameters obtained after the execution of the intended algorithm. After using optimized hyperparameters values to verify the prediction of six network models, it can be found that the accuracy is increased. The following fine-tuned parameter values were used to perform the second prediction test: No of Neurons in first layer 256, window steps 6, Epochs 1000, Batch size 64 and Learning rate 0.2. It can be observed that the prediction accuracy of various models increases by using tuned hyperparameters and tuned sliding window length obtained for GRU network. Results are presented with the gain measure obtained for all of the six models. Results show that average gain value for the optimized GRU is higher than the normal untuned model with a relative improvement of 4.5%.

Conclusion

In this paper, the fundamentals of the GRU network was presented and proposed a hyperparameter optimization analysis coupled with window steps tuning for time series prediction. The advantage of starting with a simple search and progressively narrow down subsequent searches keeping one parameter variable and others fixed was addressed. To search for better parameter sets from sample space, five experiments in series were conducted. Results show a higher capability of the proposed method to reduce the error and an average gain of the optimized network over the normal model is 4.5%. After obtaining the refined parameter sets, tests on various models in order to achieve accuracy and stability is performed. The proposed GRU parameter optimization algorithm helps to search the suitable parameter combinations that minimize the error metrics; RMSE, MAPE and MAE measure 7.13, 5.93 and 3.47 respectively. This paper further verifies the applicability and validity of the proposed model. The best value of performance gain is provided by GRU model then comes ARIMA and CNN-LSTM models with the Gain values 4.50, 3.42 and 3.29 respectively.