Large-scale and Long-term Forecasting of Performance Measurement of Public Transportation Systems

July 2021

A White Paper from the Pacific Southwest Region University Transportation Center

Luan Tran, Ph.D. Student, University of Southern California

Yao Yi Chiang, Co-Principal Investigator, University of Southern California

Cyrus Shahabi, Principal Investigator, University of Southern California

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No. PSR-20-21 TO-038	**2. Government Accession No.** N/A	**3. Recipient's Catalog No.** N/A
4. Title and Subtitle Large-scale and Long-term Forecasting of Performance Measurement of Public Transportation Systems		**5. Report Date** 7/26/2021 **6. Performing Organization Code** N/A
7. Author(s) Yao-Yi Chiang, 0000-0002-8923-0130 Cyrus Shahabi, 0000-0001-9118-0681 Luan Tran		**8. Performing Organization Report No.** PSR-20-21
9. Performing Organization Name and Address METRANS Transportation Center University of Southern California University Park Campus, RGL 216 Los Angeles, CA 90089-0626		**10. Work Unit No.** N/A **11. Contract or Grant No.** USDOT Grant 69A3551747109 Caltrans 65A0674 TO 038
12. Sponsoring Agency Name and Address California Department of Transportation (Caltrans) 1801 30th St Sacramento, CA 95816		**13. Type of Report and Period Covered** Final report (8/16/2020 – 8/15/2021) **14. Sponsoring Agency Code** N/A

15. Supplementary Notes
https://infolab.usc.edu/caltrans/index.html#/

16. Abstract
Accurate forecasting of public transportation metrics is critical towards the high reliability and efficiency of the public transportation system. However, deploying a forecasting system to serve city-level public transportation with long-term forecasting is challenging. In this project, we develop the capability for processing the entire Los Angeles Metropolitan Area (LAMA) for long-term forecasting of a variety of public transportation system performance metrics. First, we explore both spatial statistical methods and machine learning methods to estimate traffic flows for the road segments that do not have traffic sensors. Second, we develop methods to enable traffic forecasting with a deep learning model designed for small networks for the entire LAMA road network. We also study various training strategies (e.g., teacher forcing) to enable accurate long-term forecasting of traffic flows and bus arrival times. Lastly, we develop an end-to-end deep learning approach that combines the estimation and forecasting of traffic flow with data imputation methods for estimating bus arrival time for each stop in individual bus routes in LAMA. Using the real-world data in the University of Southern California Archived Transportation Data Management System (ADMS), we show that the proposed approach and system are capable of predicting bus arrival times with a city-level spatial coverage and a route-level temporal forecasting horizon. We also demonstrate the overall result of the bus arrival time estimation in a web dashboard. This dashboard enables users at all levels of technical skills to benefit from the developed machine learning approach and access to valuable information for trip planning, vehicle management, and policymaking.

17. Key Words Traffic Forecasting, Estimated Time of Arrival	**18. Distribution Statement** No restrictions.		
19. Security Classif. (of this report) Unclassified	**20. Security Classif. (of this page)** Unclassified	**21. No. of Pages** 35	**22. Price** N/A

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Contents

About the Pacific Southwest Region University Transportation Center

The Pacific Southwest Region University Transportation Center (UTC) is the Region 9 University Transportation Center funded under the US Department of Transportation's University Transportation Centers Program. Established in 2016, the Pacific Southwest Region UTC (PSR) is led by the University of Southern California and includes seven partners: Long Beach State University; University of California, Davis; University of California, Irvine; University of California, Los Angeles; University of Hawaii; Northern Arizona University; Pima Community College.

The Pacific Southwest Region UTC conducts an integrated, multidisciplinary program of research, education and technology transfer aimed at *improving the mobility of people and goods throughout the region.* Our program is organized around four themes: 1) technology to address transportation problems and improve mobility; 2) improving mobility for vulnerable populations; 3) Improving resilience and protecting the environment; and 4) managing mobility in high growth areas.

U.S. Department of Transportation (USDOT) Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by a grant from the U.S. Department of Transportation's University Transportation Centers Program. However, the U.S. Government assumes no liability for the contents or use thereof.

California Department of Transportation (CALTRANS) Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated under the sponsorship of the United States Department of Transportation's University Transportation Centers program, in the interest of information exchange. The U.S. Government and the State of California assumes no liability for the contents or use thereof. Nor does the content necessarily reflect the official views or policies of the U.S. Government and the State of California. This report does not constitute a standard, specification, or regulation. This report does not constitute an endorsement by the California Department of Transportation (Caltrans) of any product described herein.

Disclosure

Shahabi (PI), Chiang (co-PI), and Tran (PhD student), conducted this research titled, "Large-scale and Long-term Forecasting of Performance Measurement of Public Transportation Systems" at the Computer Science Department and Spatial Sciences Institute, University of Southern California. The research took place from 8/16/2020 to 8/15/2021 and was funded by a grant from Caltrans in the amount of $100,000. The research was conducted as part of the Pacific Southwest Region University Transportation Center research program.

Acknowledgments

We thank Matthew Lim and Mingxuan Yue for their valuable contribution to this project.

Abstract

Accurate forecasting of public transportation metrics is critical towards the high reliability and efficiency of the public transportation system. However, deploying a forecasting system to serve city-level public transportation with long-term forecasting is challenging. In this project, we develop the capability for processing the entire Los Angeles Metropolitan Area (LAMA) for long-term forecasting of a variety of public transportation system performance metrics. First, we explore both spatial statistical methods and machine learning methods to estimate traffic flows for the road segments that do not have traffic sensors. Second, we develop methods to enable traffic forecasting with a deep learning model designed for small networks for the entire LAMA road network. We also study various training strategies (e.g., teacher forcing) to enable accurate long-term forecasting of traffic flows and bus arrival times. Lastly, we develop an end-to-end deep learning approach that combines the estimation and forecasting of traffic flow with data imputation methods for estimating bus arrival time for each stop in individual bus routes in LAMA. Using the real-world data in the University of Southern California Archived Transportation Data Management System (ADMS), we show that the proposed approach and system are capable of predicting bus arrival times with a city-level spatial coverage and a route-level temporal forecasting horizon. We also demonstrate the overall result of the bus arrival time estimation in a web dashboard. This dashboard enables users at all levels of technical skills to benefit from the developed machine learning approach and access to valuable information for trip planning, vehicle management, and policymaking.

Large-scale and Long-term Forecasting of Performance Measurement of Public Transportation Systems

Executive Summary

Though traffic congestions have been largely reduced due to the pandemic in 2020, as Los Angeles reopens after enduring more than a year of COVID-19 restrictions, traffic congestions rapidly approach pre-pandemic levels. Hence, the enormous economic and social cost caused by traffic congestions will continue to be a major concern of the policymakers and all citizens. One solution to mitigate the congestions is an efficient public transportation system. A reliable public transportation service could potentially release drivers from their cars and improve overall citizens' mobility efficiency. Accurate forecasting of public transportation metrics is critical towards the high reliability and efficiency of the public transportation system. However, deploying a forecasting system to serve city-level public transportation with long-term forecasting is challenging. In this project, we develop the capability for processing the entire Los Angeles Metropolitan Area (LAMA) for long-term forecasting of a variety of public transportation system performance metrics. First, we explore both spatial statistical methods and machine learning methods to estimate traffic flows for the road segments that do not have traffic sensors. Second, we develop methods to enable traffic forecasting with a deep learning model designed for small networks for the entire LAMA road network. We also study various training strategies (e.g., teacher forcing) to enable accurate long-term forecasting of traffic flows and bus arrival times. Lastly, we develop an end-to-end deep learning approach that combines the estimation and forecasting of traffic flow with data imputation methods for estimating bus arrival time for each stop in individual bus routes in LAMA. Using the real-world data in the University of Southern California Archived Transportation Data Management System (ADMS), we show that the proposed approach and system are capable of predicting bus arrival times with a city-level spatial coverage and a route-level temporal forecasting horizon. Our traffic forecasting experimental results compared with a baseline approach shows that our approach significantly outperformed the baseline approach. Our approach had small prediction errors for traffic forecasting at 15, 30, 45, and 60 minutes into the future, which enables accurate estimates of bus arrival times. We also demonstrate the overall result of the bus arrival time estimation in a web dashboard. Using the dashboard, a user can select a bus route, the starting and ending bus stops, and the time to leave to the bus arrival time estimation at each bus stop along the route. This dashboard enables users at all levels of technical skills to benefit from the developed machine learning approach and access to valuable information for trip planning, vehicle management, and policymaking.

Introduction

Though the traffic has been largely reduced due to the pandemic in 2020, Los Angeles is still ranked as the most congested city in the U.S. according to TomTom Traffic Index Ranking.[1] And as Los Angeles reopens after enduring more than a year of COVID-19 restrictions, the traffic congestion is rapidly approaching pre-pandemic levels. Hence, the enormous economic and social cost caused by the congestion will continue to be one of the major concerns of the policymakers and all citizens. One potential solution to mitigate the congestion is an efficient public transportation system. A reliable public transportation service could potentially release more drivers from their cars and improve the efficiency of transportation agencies. Accurate forecasting of public transportation metrics is a critical pathway towards high reliability and efficiency.

In this project, we build on our prior approach and system to develop the capability for processing the entire Los Angeles Metropolitan Area (LAMA) for long-term forecasting of a variety of public transportation system performance metrics. The major challenges include 1) how to predict/infer traffic speed on a road segment when there is no traffic sensor at the road segment, 2) how to overcome the computational and memory bottlenecks of training a deep learning network for traffic flow forecasting over a large road network, and 3) how to deal with the large error propagation in the deep learning network for forecasting over a long-time span. We explore both spatial statistical methods and machine learning methods for the estimation of traffic speed for the road segments that do not have a traffic sensor. We investigate methods to represent large road networks into a condensed graph or multiple subgraphs so that the deep learning network can scale up to process large spatial areas. We also study various training methods (e.g., teacher forcing) to enable long-term forecasting of both traffic flows and the performance measurement of public transportation systems. Using real-world freeway and arterial traffic data and bus location data, we showed that the proposed system is capable of predicting bus arrival times with a city-level coverage and a route-level forecasting horizon. To demonstrate the results from the proposed research, we also develop a web application in which users can select the start time of a trip and retrieve the predicted bus arrival times anywhere in the entire Los Angeles Metropolitan Area.

Specifically, the project includes the following six tasks:

- Task 1: Traffic Prediction for Bus Performance Metrics Estimation on Local Streets
- Task 2: Graph Partitioning for Large-scale Traffic Forecasting
- Task 3: Teacher Forcing for Long-term Forecasting
- Task 4: Integration and Testing of all modules
- Task 5: Web Dashboard Development

[1] https://www.tomtom.com/en_gb/traffic-index/ranking/?country=US

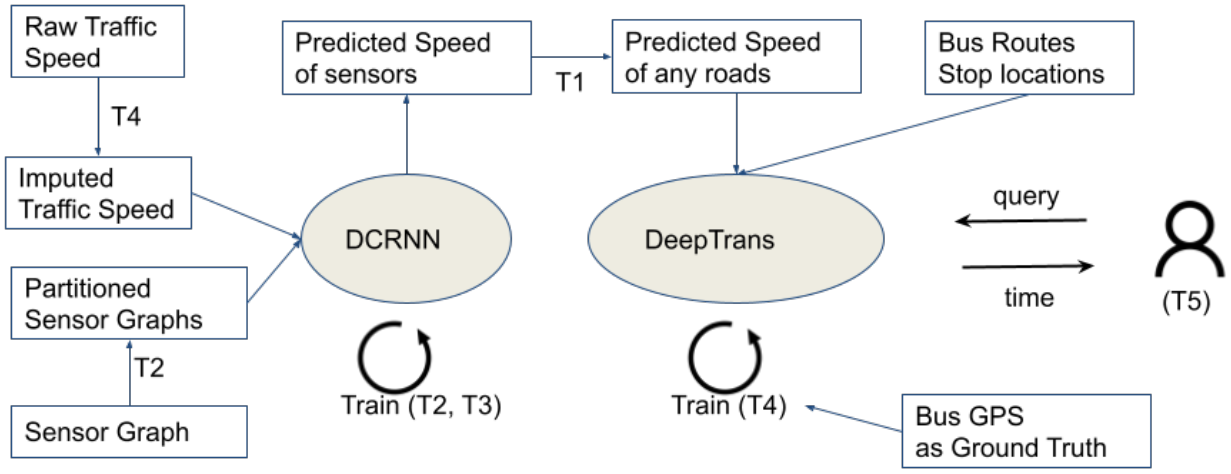
- Task 6: Final report

We describe the workflow of our framework in Figure 1 with the notations of corresponding tasks. In detail, there are mainly two deep learning models integrated into the framework: Diffusion Convolutional Recurrent Neural Network (DCRNN) (for traffic forecasting) and DeepTrans (for bus arrival estimation).

- **DCRNN:** The inputs of DCRNN include the raw traffic speed collected from loop sensors on the roads and the graph of sensors. The raw traffic data are imputed since they usually contain missing data (in Task 4). The sensor graph is partitioned for efficient training of DCRNN (in Task 2). The traffic forecasting model DCRNN is trained with advanced architectures (in Task 2 and 3). The output of DCRNN is the predicted speed at sensor locations which is used to estimate the speed of any road segments (in Task 4).
- **DeepTrans:** The estimated speed at all road segments and the bus route information are the inputs to DeepTrans, which is trained using the Bus GPS data as the ground truth for arrival time estimation (in Task 4).

Finally, the trained DCRNN and DeepTrans support a frontend interface, which can process user queries and show the predicted results in a web dashboard (in Task 5). The remaining sections of the report will introduce these tasks following the same order as the task number.

Figure 1. Framework structure and tasks



Traffic Prediction for Bus Performance Metrics Estimation on Local Streets

Traffic flow information is crucial in predicting bus arrival time. Although the speed values at thousands of sensors are available for LAMA, buses often travel on roads that do not have traffic

sensors. Therefore, traffic flow prediction at non-sensor locations is crucial. We formalize the problem as follows:

Problem Definition: *Given a set of locations with traffic speed values, we want to predict/estimate the speed values at a set of nearby locations.*

Method

Since traffic flows generally follow the first law in geography, it is a natural choice to use the K Nearest Neighbor to estimate speed for a location from nearby sensors. Particularly, for each non-sensor location, we take the average speed values from its K nearest sensors weighted by various distance metrics.

Assume location l_i has K nearest neighbors s_1, s_2, …, s_K with the distances d_1, d_2, …, d_K, the speed at l_i is computed as:

$$v(l_i) = \sum_{k=1}^{K} w(d_k) * v(s_k)$$

with $w(d_k)$is a weight function based on the distance from l_i to s_k, i.e., d_k .
In this study, we examine two weight functions:

1. **Uniform Weight:** $w(d_k) = 1/K$ for all k,
2. **Inverse Distance Weight:** $w(d_k) = (1/d_k)/(1/d_1 + 1/d_2 + \ldots + 1/d_K)$

Specifically, for cases using Inverse Distance Weight, we consider two distance functions: Euclidean distance (EU) and driving distance.[2]

Experiment Results

We compared our prediction methods using two kinds of traffic sensors: 1) 11,126 arterial sensors, and 2) 1,192 highway sensors. For each sensor, we predict its speed value based on its K nearest neighbors and report the Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). Tables 1 and 2 report the RMSE and MAPE with the arterial sensors. As can be seen in these tables, distance (EU and driving distances) weighted functions offer lower RMSE and MAPE than the uniform weight function. Tables 3 and 4 report the RMSE and MAPE with the highway sensors. As reported, the driving distance offers the lowest error. In this scenario, the EU distance weighted function produces more errors because highway sensors closed in EU distance can be located on two different highways (or highway directions).

[2] We retrieve the road network distance from the Google routing API.

Table 1. RMSE of arterial estimation with different K

Method	K = 3	K = 4
Weighted based on EU distance	5.85	5.73
Uniform weight	6.21	6.1
Weighted based on driving distance	5.86	5.71

Table 2. MAPE of arterial estimation with different K

Method	K = 3	K = 4	K = 5
Weighted based on EU distance	33.0	32.6	32.4
Uniform weight	37.1	35.6	35.2
Weighted based on driving distance	32.9	32.7	32.8

Table 3. RMSE of highway estimation with different K

Method	K = 3	K = 4	K = 5
Weighted based on EU distance	9.35	9.26	9.16
Uniform weight	8.67	8.46	8.35
Weighted based on network distance	8.38	8.28	8.19

Table 4. MAPE of highway estimation with different K

Method	K = 3	K = 4	K = 5
EU distance Weight	12.16	12.04	11.95
Uniform weight	11.42	11.21	11.16
Weighted based on network distance	10.75	10.73	10.69

Conclusion

We examined the K Nearest Neighbor approach to predict the traffic flows at non-sensor regions. We showed that for arterial sensors, Euclidean and driving distances offered comparable accuracy, while for highway sensors, driving distances performed best.

Graph Partitioning for Large-scale Traffic Forecasting

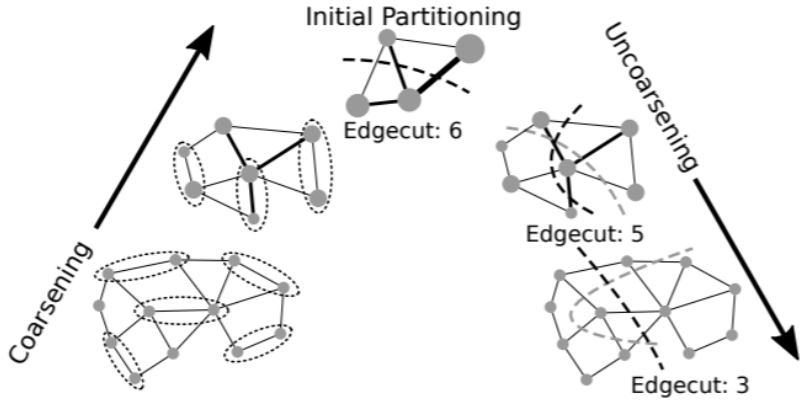
To train a traffic flow forecasting model, we employ the DCRNN model [2] that takes the speed values at a set of sensors in the previous T timestamps and the sensors' distance matrix as input. DCRNN uses an encoder-decoder framework that captures the spatial and temporal dependency between sensors. The traditional pipeline of using all the data to train a global traffic model is not feasible for LAMA for two reasons. First, the training data of more than 10,000 sensors in LAMA, is too large to fit in the computer memory. For example, forecasting on 300 sensors requires 11 GB of memory. Second, the training time for such a large dataset becomes prohibitive.

Method

To overcome this challenge of large road networks, we adopt a graph-based partitioning method and divide the forecasting problem into small forecasting sub-problems. The intuition behind this is that groups of sensors that are very far from each other do not directly affect each other.

The first step of training DCRNN is the construction of the sensor network graph. In the sensor network graph, each node represents a traffic sensor, and edge weights represent the spatial distances between sensors. For each sensor, we first identify its K nearest neighbors in Euclidean space and then compute the road network distance from the sensor to its neighbors to construct the edge weights. Next, we partition the graph into small subgraphs. A straightforward method is to partition the sensor network graph using K-means clustering. However, this method often leads to unbalanced clusters. Ideally, we would like to have partitions of the same size so that the average training time per partition is minimized. Therefore, we adopt Metis [3], which is a multi-level K-way partitioning algorithm for network partitioning. Metis produces K clusters of roughly the same size by iteratively collapsing groups of connected nodes into "supernodes". Specifically, Metis first coarsens a large graph by merging nodes iteratively (Figure 2). And then starting from the coarsest graph, Metis partitions and reverses the coarsening operation to obtain the final partitioning results.

Figure 2. Metis Partitioning Process [3]



Lastly, we train a traffic forecasting model for each cluster (subgraph). If the number of clusters is large, then each cluster becomes too small, and we lose the information regarding the spatial dependencies between sensors. Conversely, if the number of clusters is small, then clusters become too large, and the time and resources required for training become prohibitive. In our dataset, the optimal number of clusters is 15, with each partition containing slightly more than 600 sensors.

Experiment Results

In this work, we consider the Historical Average (HA), Vector Autoregressions (VAR), and DCRNN models for traffic forecasting. We believe that HA is what most industries use, VAR is the state-of-the-art academic traffic forecasting model, and DCRNN is our own approach. We used a dataset of 10,000 sensors over a period of six months (70% for training, 10% for validation, and 20% for testing). For VAR and DCRNN, we split the sensor network into 15 partitions of roughly the same size using the approach described in the previous section. Table 5 shows the Mean Absolute Error (MAE) in miles/hour at 15, 30, 45, and 60 minutes into the future. HA estimates are independent of time hence its MAE is independent of the prediction horizon. Long-term traffic prediction is challenging due to unforeseen incidents hence the error of VAR and DCRNN increases with the prediction horizon. DCRNN's MAE is consistently lower than VAR's by a factor of up to ~2.4 because the former can effectively capture the spatial dependencies between traffic sensors.

Table 5. Models Mean Absolute Error (MAE) in miles/hour at 15, 30, 45, and 60 minutes.

Model	15 min	30 min	45 min	60 min
HA	9.94	9.94	9.94	9.94
VAR	7.24	8.66	9.46	10.06
DCRNN	3.25	3.73	4.02	4.24

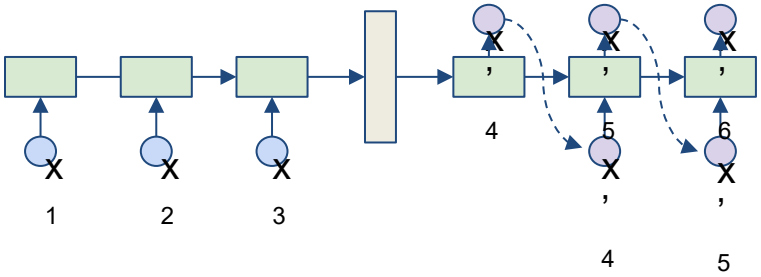
Teacher Forcing for Long-term Forecasting

Long-term traffic forecasting is challenging due to unforeseen incidents. Like the majority of the state-of-the-art machine learning models for time-series data forecasting, our DCRNN model uses recurrent neural networks (RNNs), in which errors can be accumulated over time and makes the model difficult to converge.

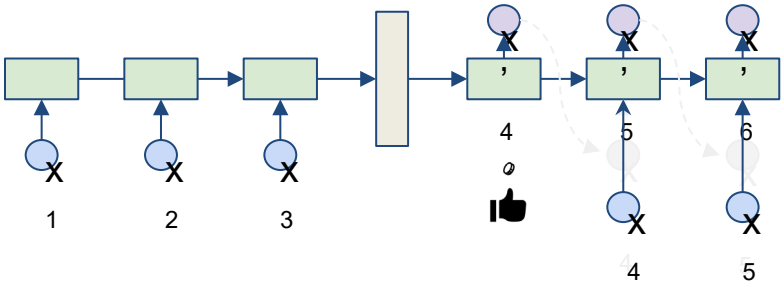
Method

To overcome this challenge, we adopt the teacher forcing technique with scheduled sampling for quickly and efficiently training the recurrent neural networks. Figure 3 compares RNNs with and without teacher forcing. Without using teacher forcing, RNNs will use their predicted output from the previous step as the input of the current step. Thus, the error will accumulate. In Figure 2 (b), RNNs with teacher forcing use the ground truth of the previous step as the input of the current step. Therefore, even if the model produces a large error in the previous step, the error will not accumulate into the next steps. However, as the inference process does not have the ground truths for future steps and still needs to follow the procedure in Figure 2 (a), RNNs learned from teacher forcing may overfit the training dataset. Therefore, we employ a scheduled sampling strategy, which means the model decides whether or not to use teacher forcing by some decaying probability. With this strategy, as the training process progresses, the model involves less teacher forcing in the training.

Figure 3. Training with (without) teacher forcing in a recurrent neural network



(a) Training without teacher forcing



(b) Training with teacher forcing by scheduled sampling

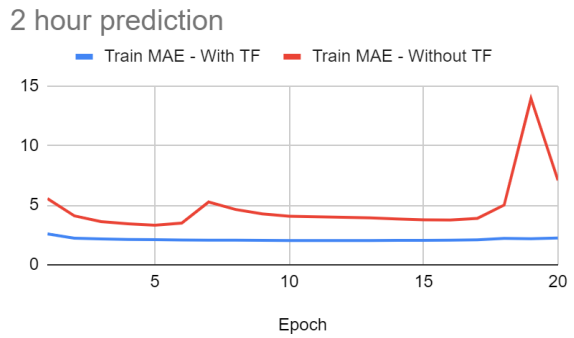
Experiment Results

We compared the MAE of the DCRNN model using a group of arterial sensor networks of 745 sensors in predicting the traffic speed in the next one and two hours with and without using teacher forcing. Table 6 reports the MAE of the DCRNN model after training 20 epochs. For both cases, employing the teacher forcing technique resulted in a lower error compared to the original DCRNN. Figure 4 depicts the training loss of the model during 20 epochs. The training loss converged quickly using teacher forcing. While without the teacher forcing technique, the training loss did not converge within 20 epochs.

Table 6. DCRNN's Mean Absolute Error (MAE) With and Without Teacher Forcing

MAE	DCRNN+Teacher Forcing	DCRNN
1 hour ahead prediction	3.086	3.1007
2 hour ahead prediction	**3.6102**	4.19

Figure 4. Training Loss with and without Teacher Forcing (TF)

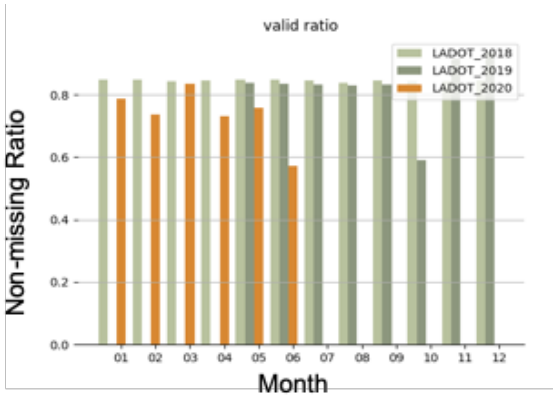


Integration and Testing of all modules

Traffic Flow Imputation

The accuracy of traffic flow forecasting highly depends on the accuracy and quality of the collected data, i.e., the collected sensor speed. However, missing data remain a challenge in many current traffic data management systems. For example, in our ADMS system, we encountered 40% missing traffic speed values in June 2020 (Figure 5).

Figure 5. The non-missing ratio of speed values in the ADMS system until June 2020



A number of methods have been introduced for imputing missing values. In this study, we compare four state-of-the-art approaches including 1) **Historical** which uses the average value in past 2 weeks at the same hour, 2) **K Nearest Neighbors (KNN)** which uses the non-missing values at the K nearest neighbors to the missing location, 3) **Graph Convolutional Neural Network Autoencoder (GCNN)** which learns the representation of traffic network, 4) **Low-Rank Matrix Decomposition (LRMD)** which learns the low-rank matrices that decomposes the matrix representing time-series data of speed at multiple locations.

We compare the imputation accuracy of these methods using traffic flow data we collected from thousands of sensors located in Los Angeles. We observe that LRMD outperforms the other methods when the missing ratio is small while Historical performs best when the missing ratio is large.

Problem and Method

Problem Formulation. Given a set of N traffic sensors S = {s^1, s^2, …, s^N} at the most recent T timestamps. Each sensor s^i has T values of speed denoted as s^i_1, s^i_2, …, s^i_T. At each timestamp t, if the speed value s^i_t exists, it is a non-negative number. If there is a missing value at time t for sensor s^i, s^i_t = -1. We want to impute the missing values at positions where s^i_t = -1.

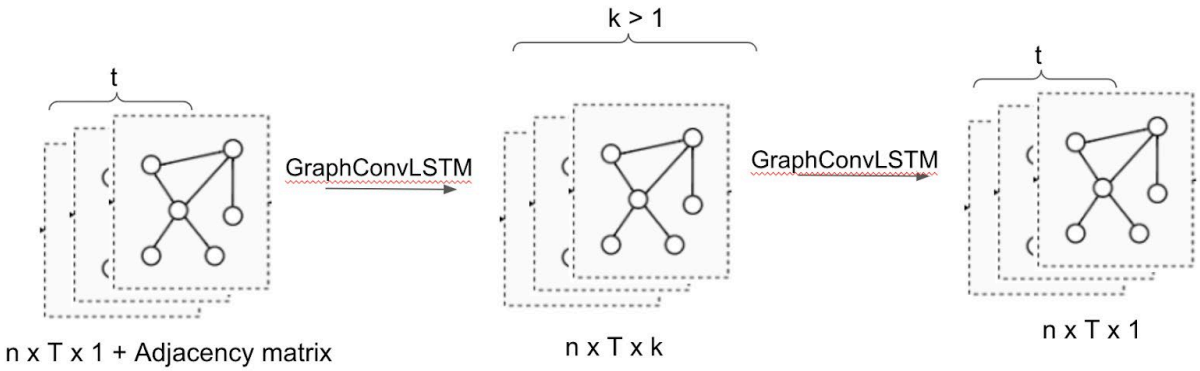
Historical. This approach employs historical data to impute missing values. Specifically, we impute the missing values by the average value at the same sensor and same hour in the previous 2 weeks.

K Nearest Neighbors (KNN). This approach imputes the missing values using the weighted average values at the neighboring sensors. Specifically, for each sensor s^i, it computes the closest K neighboring sensors with values v_1, v_2, …, v_K, and v_j >=0 for any 1<=j<=K. Assume the distances from s^i to their neighbors are d_1, d_2, .., d_K. The imputed value at s^i is as follows:

$$\text{imputed}(s^i_t) = \left(\sum_{j=1}^{K} v_j/d_j\right)/\left(\sum_{j=1}^{K} 1/d_j\right)$$

Graph Convolutional Neural Network Autoencoder (GCNN). This approach takes the input matrix (each row is a time-series speed value of one sensor) and a graph whose nodes are traffic sensors and edges are distances between sensors. It employs an autoencoder to learn the representation of the network and then decoded results will be used as the imputed values (Figure 6). The encoder and decoder consist of GraphConvolutionLSTM layers to capture the spatial and temporal dependency between traffic sensors.

Figure 6. GCNN model



Low-Rank Matrix Decomposition (LRMD). [4] This approach decomposes the matrix M of speed values of N sensors and T times into two low-rank matrices U and V.

$$M \approx U^{\top}V$$

LRMD optimizes U and V with low-rank constraints. Specifically, U has a shape of N x k and V has a shape of k x T. The matrix U captures the temporal dependency and the matrix V captures the spatial dependency. Assuming after running optimization we obtain U' and V', the imputed matrix can be calculated as $U'^{\top}V'$

Experiment Results

We collected the speed values in one day in 1,837 traffic sensors on highways in Los Angeles. Speed data are aggregated for every hour. We compared all approaches with two metrics:

MAPE = (1/n) * Σ(|actual – predicted| / actual) * 100

RMSE = sqrt(Σ((actual – predicted)^2 / n))

where n is the number of sensors; the actual is the ground-truth value, and the predicted is the imputed value.

We examined two scenarios: random missing and consecutive missing. For random missing, we randomly selected the positions and times to obtain missing values. Their true values are replaced by -1. For consecutive missing, we randomly selected the sensor locations and some consecutive periods, based on specified missing rates and replaced all values in the consecutive periods with -1.

Random Missing. Tables 7 and 8 report the RMSE and MAPE for the random missing scenario with the missing rate varied from 20% to 60%.

Table 7. RMSE for the random missing scenario.

Missing Rate	20%	40%	60%
KNN	9.69	9.82	9.86
GCNN	10.25	10.33	10.39
LRMD	7.29	7.36	7.61
Historical	8.9	8.88	8.91

Table 8. MAPE for the random missing scenario (%).

Missing Rate	20%	40%	60%
KNN	14.82	14.92	15.04
GCNN	16.6	16.63	16.69
LRMD	10.09	10.06	10.41
Historical	11.88	11.64	11.67

As reported in these tables, LRMD offered the lowest RMSE and MAPE for all test cases. The reason is that LRMD could automatically learn the correlations between traffic sensors. GCNN produced the worst performance since it would need large data to train the neural networks. Historical only used the historical statistics and hence could not capture the current status of the traffic network.

Consecutive Missing. Tables 9 and 10 report the RMSE and MAPE for consecutive missing scenarios when the missing rate is varied from 20% to 60%.

Table 9. RMSE for the consecutive missing scenario.

Missing Rate	20%	40%	60%
KNN	9.72	10.19	10.23
GCNN	10.52	10.53	10.53
LRMD	7.29	7.9	33.05
Historical	8.94	9.26	9.25

Table 10. MAPE for the consecutive missing scenario.

Missing Rate	0.2	0.4	0.6
KNN	14.59	15.78	15.75
GCNN	16.63	16.69	16.61
LRMD	9.79	11.02	35.72
Historical	12.1	12.31	12.31

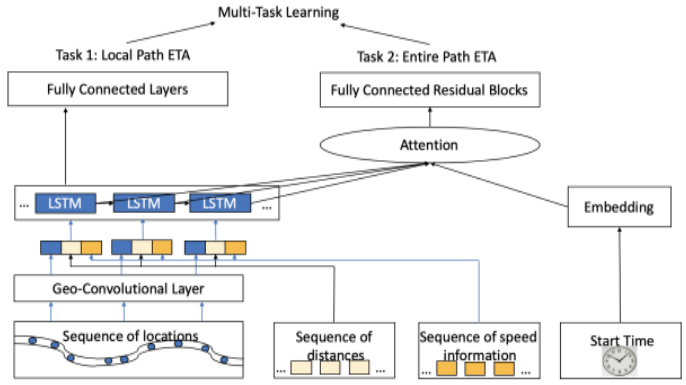
LRMD offered the lowest errors for low missing rates. In these cases, LRMD could effectively learn the dependencies between sensors. While in the cases of high missing rate (60%), LRMD failed to learn those dependencies, and Historical produced the lowest errors.

Overall Bus Arrival Time Estimation

Problem and Method

In this study, we investigate combining the predicted traffic flow with the Bus ETA (estimated time of arrival) model [1]. Figure 7 shows the network architecture of the Bus ETA model. The model takes sequences of bus stop locations (longitude and latitude), the predicted speed values at those locations, a starting time & location, and distances from those bus stops to the starting location as input. The output is the predicted arrival time at each bus stop on the bus route.

Figure 7. Bus ETA Model Architecture



The predicted speed at bus stop locations would affect the estimation of bus arrival time, and the estimation of arrival times at a stop also determines the predicted speed. Thus, we consider several approaches in preparing the sequence of predicted speeds at future bus stops as follows.

No speed. In this approach, we do not utilize the speed information at each bus stop.

Ground-truth 1 hour ahead. In this approach, we assume an ideal scenario, in which we can have the true speeds at each location in the next 1 hour. This is not a real scenario but is the best-case scenario.

Ground-truth for the entire route. There are bus routes that take more than 1 hour. We assume an ideal scenario that we have the true speeds at each location for the entire bus route. Similarly, this is not a real scenario, but with this, we can obtain the best-case scenario.

Historical Speed + Traffic Forecaster. In this approach, we compute the average speed for each bus route in the historical data. Then, for each bus route, we compute the estimated arrival time for each bus stop based on its average speed. After obtaining the rough arrival times at all stops, we use our traffic forecasting model to estimate the speed at each location at the estimated arrival time at each bus stop. Then we use the predicted speeds to calculate the final ETA.

Iterative Traffic Forecasting. In this approach, we define a starting speed as the predicted traffic speed at the starting location at the starting time. The starting speed is used to compute the estimated arrival time of the following bus stop. Then with the estimated time, we use the traffic forecasting model to obtain the predicted speed at the next stop given the estimated time. Similarly, we iteratively compute the predicted speed and the arrival times at the remaining bus stops.

Experiment Results

We compared the Mean Absolute Percentage Error (MAPE) of all the approaches using 5,400 bus routes in LAMA in a week in 2017. The dataset was split into train, validation, and test data with the ratio of 7:1:2. Table 11 reports the MAPE for the training, validation, and test datasets. Using the Iterative Traffic Forecasting approach produced the closest results to the ground truth for the entire route (i.e., the best-case scenario). The results also confirm that our approach for forecasting traffic flow is effective. Because the Iterative Traffic Forecasting approach had the best performance, we incorporated it as part of our end-to-end system, DeepTrans, for large-scale and long-term bus arrival time estimation.

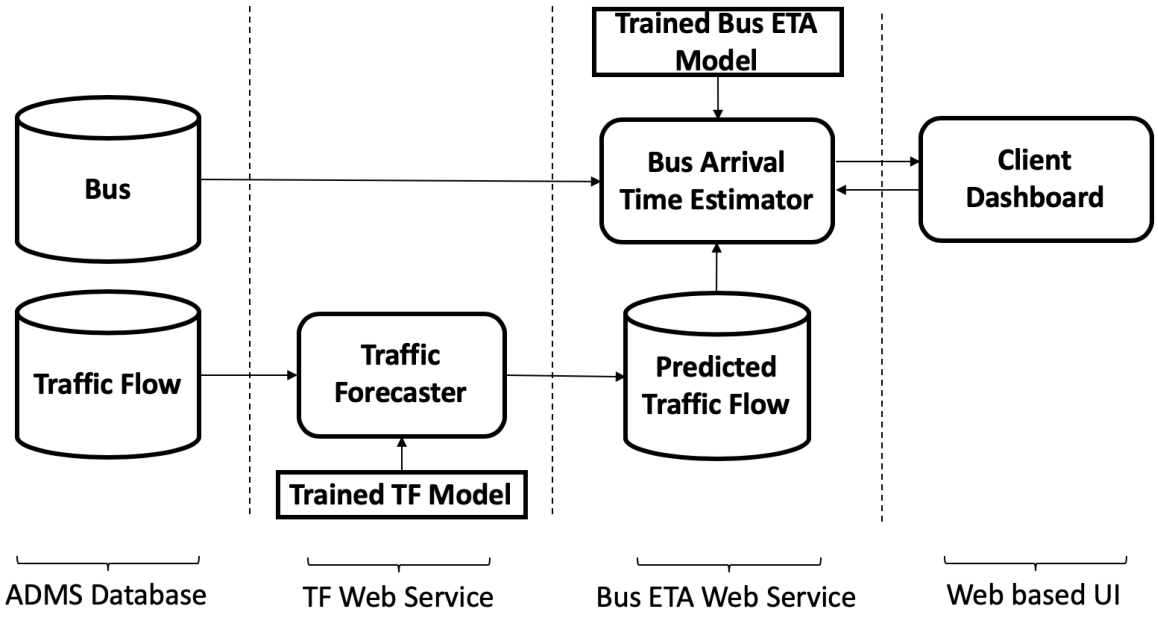
Table 11. MAPE of approaches

Approach	Train	Val	Test
No speed	0.174	0.316	0.31
Ground-truth 1 hour ahead	0.156	0.259	0.262
Ground-truth for Entire Route	**0.146**	**0.227**	**0.238**
Historical Speed + Traffic Forecaster	0.162	0.258	0.252
Iterative Traffic Forecaster	**0.15**	**0.23**	**0.24**

Web Dashboard Development

To demonstrate the bus arrival time estimation system, we develop both the front-end dashboard and back-end services of the system as illustrated in Figure 7.

Figure 8. DeepTrans overview



ADMS Database. With our partnership with the Los Angeles County Metropolitan Transportation Authority (LACMTA), we developed a large transportation data warehouse called Archived Traffic Data Management System (ADMS). The ADMS Database stores the bus and traffic flow information. This dataset includes both sensor metadata and real-time data for freeway and arterial traffic sensors (approximately 16,000 loop-detectors) covering 4,300 miles and 2,000 bus and train automatic vehicle locations with an update rate as high as 30 seconds. We have been continuously collecting and archiving the datasets mentioned above for the past 12 years. These datasets are used to train and test our traffic forecasting and Bus ETA models.

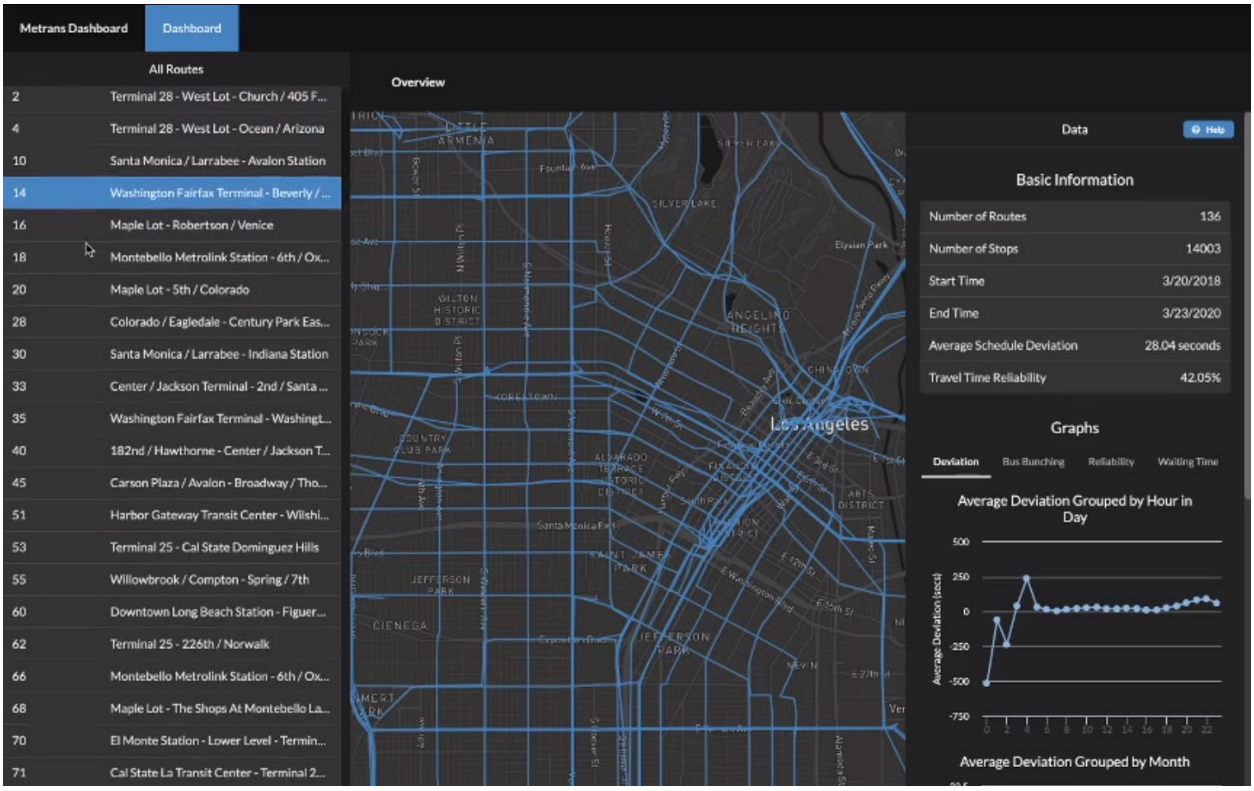
Traffic Forecasting Service. We deployed the DCRNN model to a web service to predict traffic flow in the next several hours.

Bus Arrival Time Estimator. We deployed the bus ETA model to a web service to predict the time of arrival for bus stops given starting time, starting stop, and the bus routes.

Web-based UI. We developed a Web-based User Interface that presents the information of bus routes and bus stops and allows users to plan a trip in the future with the predicted bus arrival times at every bus stop on a selected bus route.

Figure 9 presents the user interface of our end-to-end system, DeepTrans. It shows the front page with the overview information of all bus routes in Los Angeles. The left sidebar contains a table displaying every bus route's name, and the right sidebar shows the statistical information of all bus routes. The map at the center shows all bus routes. It allows the user to zoom in or out of the map view to explore different levels of details. When a bus route from the left sidebar is selected, the map updates to show only the selected route. The left-hand table updates to display the list of the bus stops in the route, and the right sidebar shows the statistics of the selected bus route. The user can select the start and destination bus stations and the start time of travel. Then by clicking the "Estimate" button, the dashboard sends requests to the Bus ETA Web Service and displays the estimated arrival times for each bus stop in the left-hand table. The center map reflects the table and only shows the route from the start to the end bus stations. When the user clicks each bus stop in the left-hand table, the right sidebar also shows the statistical information specific to the bus stop.

Figure 9. Frontpage UI

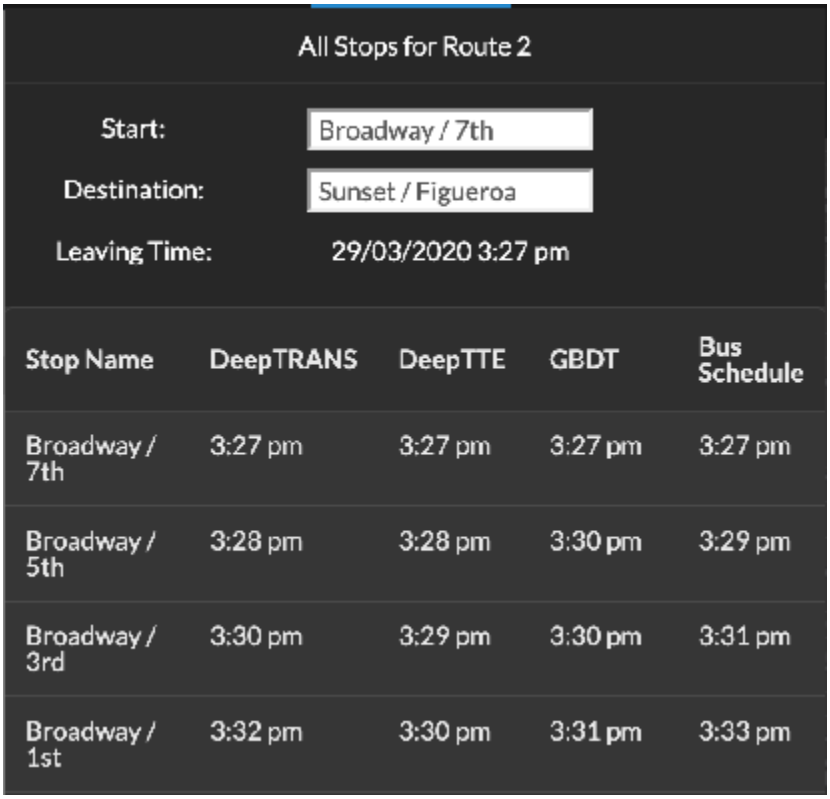


DeepTRANS can be used for everyday travel. For example, a user may be planning their commute from their home to their work office. The user can select a bus route, followed by the starting and ending bus stops. Then they can select the time to leave or keep the leaving time with the default option “Leave Now”, which takes the nearest bus schedule of the chosen starting bus stop as the leaving time. Consequently, the user will obtain the ETA at each bus stop along the route. Thus, the user can plan their trip ahead of time. Alternatively, a user can also verify the performance of DeepTRANS by checking the algorithms’ ETA for past data by comparing them to ground truth arrival times if available.

We provide demonstrations for these two scenarios: first, when a user selects a start time in the past, and second when a user selects a start time in the future. For both cases, we show the estimated times of arrival from our Bus ETA model, DeepTrans, and another deep learning model, DeepTTE, and a statistical model, GBDT.

Scenario 1: Current or Future Bus Arrival Time Estimation. Before a user starts their trip, they select which bus route they want to take. By checking the bus arrival times estimated from DeepTRANS using different route choices and their statistical information, they decide to take Route 2. Figure 10 shows different estimated times of arrival from DeepTRANS, DeepTTE, GBDT, and Bus Schedule for each bus stop from Broadway/7th to Sunset/Figueroa for Route 2. Since the start time was chosen to be in the future, ground truth values are not available.

Figure 10. Current or Future Bus Arrival Time Estimation.



Stop Name	DeepTRANS	DeepTTE	GBDT	Bus Schedule
Broadway / 7th	3:27 pm	3:27 pm	3:27 pm	3:27 pm
Broadway / 5th	3:28 pm	3:28 pm	3:30 pm	3:29 pm
Broadway / 3rd	3:30 pm	3:29 pm	3:30 pm	3:31 pm
Broadway / 1st	3:32 pm	3:30 pm	3:31 pm	3:33 pm

Scenario 2: Bus Arrival Time Estimation in the Past. A user wants to check if the estimated times from DeepTRANS are trustworthy. To check the actual performance of DeepTRANS, the user sets the start time of a trip in the past. Figure 11 shows the ground truth of bus arrival times to all bus stops from Broadway/7th to Sunset/Figueroa of Route 2 with the start time 12/10/2019 12:07 PM, along with the four aforementioned methods of calculating times of arrivals. After exploring the results for different choices, the user finds that the estimated times of arrival from DeepTRANS are closest to the ground-truth information.

Figure 11. Bus Arrival Time Estimation in the Past

All Stops for Route 2

Start: Broadway / 7th

Destination: Sunset / Figueroa

Leaving Time: 19/12/2019 12:07 pm

Stop Name	DeepTRANS	DeepTTE	GBDT	Bus Schedule	Ground Truth
Broadway / 7th	12:07 pm	12:07 pm	12:07 pm	12:07 pm	12:07 pm
Broadway / 5th	12:08 pm	12:08 pm	12:10 pm	12:09 pm	12:09 pm
Broadway / 3rd	12:10 pm	12:09 pm	12:10 pm	12:11 pm	12:11 pm
Broadway / 1st	12:12 pm	12:11 pm	12:11 pm	12:13 pm	12:12 pm

Conclusion

In summary, we developed an approach and deployed a system for large-scale, long-term estimation of bus arrival time for LAMA. The approach is capable of handling large road networks and can provide accurate long-term traffic forecasting to enable bus arrival time estimation, even with long travel time (e.g., one hour). We also built and deployed a web dashboard, enabling users at all levels of technical skills to benefit from the developed machine learning approach and access to valuable information for trip planning, vehicle management, and policymaking. For example, the dashboard provides a unified entry point for vehicle management teams and policymakers to review historical and future bus arrival time for each route. The accurate predicted bus arrival time also encourages riders to use public transportations and in turn, help relief traffic volumes. In future work, we plan to incorporate multi-modality traffic related data sources to help improve the forecasting results. For example, we would like to incorporate event information to help capture the traffic patterns before and after large events, such as the USC football games.

References

[1] Tran, L., Mun, M. Y., Lim, M., Yamato, J., Huh, N., & Shahabi, C. (2020). DeepTRANS: a deep learning system for public bus travel time estimation using traffic forecasting. *Proceedings of the VLDB Endowment*, *13*(12), 2957-2960.

[2] Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2017). Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*.

[3] Karypis, George, and Vipin Kumar. "METIS--unstructured graph partitioning and sparse matrix ordering system, version 2.0." (1995).

[4] Chen, X., & Sun, L. (2019). Bayesian temporal factorization for multidimensional time series prediction. *arXiv preprint arXiv:1910.06366*.

Data Management Plan

Products of Research

Our partnerships with LACMTA enables our data repository, ADMS, to access large and high-resolution (both spatial and temporal) traffic sensor data from a number of transportation authorities in Southern California. This dataset includes both sensor metadata and real-time data for freeway and arterial traffic sensors (~16,000 loop-detectors) covering 4,300 miles, 2,000 bus and train automatic vehicle locations (AVL), and ramp meters with an update rate as high as 30 seconds. The dataset also includes data about several incident types (such as accidents, traffic hazards, and road closures) reported by LAPD and CHP at a rate of approximately 400 incidents per day. We have been continuously collecting and archiving the aforementioned datasets for the past 12 years and with an annual growth of 1.5TB, ADMS is the largest traffic sensor data warehouse in Southern California.

Data Format and Content

We detail the schema and the attribute of each database table in our repository.

Table Name: CONGESTION_INVENTORY

- Congestion: Data and metadata related to the road network and the congestion
- Metadata: Static information about the sensors, i.e., loop detectors

Column	Description
AGENCY(string)	Agency that provided the record
CITY(string;nullable)	City name
DATE_AND_TIME(timestamp)	Date and time
LINK_ID(string)	Sensor ID
LINK_TYPE(enum)	Sensor type (HIGHWAY, ARTERIAL)
ON_STREET(string)	Street that sensor is on
FROM_STREET(string;nullable)	The beginning street name
TO_STREET(string;nullable)	The end street name
START_LOCATION(geometry)	Latitude and longitude of sensor
DIRECTION(enum)	NORTH, SOUTH, EAST, WEST, UNSPECIFIED
POSTMILE(float)	Distance from a specific end of the road
NUM_LANES(integer)	Affected number of lanes

TableName: CONGESTION_DATA

- Data: Time-series data of each sensor; they contain the occupancy, volume, and speed for each sensor at every timestep.

Column	Description
AGENCY(string)	Agency that provided the record
DATE_AND_TIME(timestamp)	Date and time
LINK_ID(sting)	Sensor ID
OCCUPANCY(float)	The percentage of time a sensor detects a vehicle in 30 seconds ● For example, an occupancy of 5% means that of those 30 seconds, vehicle presence was detected for anaggregate1.5seconds
SPEED(float)	Distance Traveled Per Unit Time, and traffic operations ● meanspeedswithinagivenroadwaysection(link)
VOLUME(integer)	Represents the number of vehicles that passed by persensor every 30 seconds
HOVSPEED(float)	Speed of HOV lane(s)
LINK_STATUS(enum)	Status of sensor, “OK”, “FAILED” or “UNKNOWN”.

TableName: RAMP_METER_INVENTORY

- Data: Complementary to the congestion data; ramp meters are the entry points to highways from arterial roads.
- Metadata: Static information about ramp meters

Column	Description
RAMP_ID(integer)	Unique ramp id
AGENCY(string)	Agency that provided the record
DATE_AND_TIME(timestamp)	Date and time

CITY(string;nullable)	City name
MS_ID	Unique ID of the
RAMP_TYPE(integer)	The type of the ramp
ON_STREET(string)	Street that ramp is on
FROM_STREET(string;nullable)	The beginning street name
TO_STREET(string;nullable)	The end street name
LOCATION(geometry)	Latitude and longitude of sensor
DIRECTION(enum)	NORTH,SOUTH,EAST,WEST,UNSPECIFIED
POSTMILE(float)	Distance from a specific end of the road

TableName:RAMP_METER_DATA

- Data: Time-series data and state of each ramp meter at each timestamp

Column	Description
AGENCY(string)	Agency that provided the record
DATE_AND_TIME (timestamp)	Date and time
RAMP_ID(sting)	Sensor ID
DEVICE_STATUS(integer)	The device status
METER_STATUS(integer)	The meter status
METER_CONTROL_TYPE (integer)	The meter's control type
METER_RATE(integer)	The meter's measuring rate
OCCUPANCY(float)	The percentage of time a sensor detects a vehicle in 30 seconds ● For example, an occupancy of 5% means that of those 30 seconds, vehicle presence was detected for an aggregate 1.5 seconds

SPEED(float)	Distance Traveled Per Unit Time, and traffic operations ● mean speeds within a given roadway section(link)
VOLUME(integer)	Represents the number of vehicles that passed by persensor every 30 seconds
LINK_IDS(string[])	The unique ID of the road link
LINK_TYPES(string[])	The type of the road link
LINK_OCCUPANCIES (float[])	The percentage of time a sensor detects a vehicle in 30 seconds ● For example, an occupancy of 5% means that of those 30 seconds, vehicle presence was detected for anaggregate1.5seconds
LINK_SPEEDS(float[])	Distance Traveled Per Unit Time, and traffic operations ● mean speeds within a given roadway section(link)
LINK_VOLUMES(integer[])	Represents the number of vehicles that passed by persensor every 30 seconds
LINK_STATUSES(enum[])	Status of a sensor, “OK”, “FAILED” or “UNKNOWN”.

TableName: TRAVEL_TIMES_INVENTORY

- Data: Computed travel times between sensors
- Metadata: Defines the pair of sensors for which travel times were computed

Column	**Description**
LINK_ID(integer)	Unique link id
AGENCY(string)	Agency That Provided The Record
DATE_AND_TIME(timestamp)	Date And Time
ROUTE_ID(integer)	Unique Route id
DIRECTION(enum)	NORTH,SOUTH,EAST,WEST,UNSPECIFIED

LINK_TYPE(string)	Commonly Freeway
BEGIN_ID(integer)	Id of begin link
BEGIN_STREET_NAME(string)	The beginning street name
BEGIN_LOCATION(geometry)	The beginning point location
END_ID(integer)	Id of end link
END_STREET_NAME(string)	The end street name
END_LOCATION(geometry)	The end point location
LENGTH(double)	Length of link in Kilometers.

TableName:TRAVEL_TIMES_DATA

- Data: Travel time in minutes and average speed for each link pair

Column	Description
LINK_ID(integer)	Unique link ID
AGENCY(string)	Agency that provided the record
DATE_AND_TIME(timestamp)	Date and time
LINK_SPEED(double)	Average link speed in MPH
LINK_TRAVEL_TIME(double)	Estimated time to travel on the link in minutes.

TableName: BUS_INVENTORY

- Data: Static and real-time information about buses
- Metadata: Description of bus routes

Column	Description
ROUTE_ID(integer)	Unique Bus Route Id.
AGENCY(string)	Agency That Provided The Record
DATE_AND_TIME(timestamp)	Date And Time
ROUTE_DESCRIPTION(string)	Textual Description Of Route.

ZONE_NUMBERS(int[])	Ids of zones that this route operates in.

TableName:BUS_DATA

- Data: Tracking Information for buses operating on routes

Column	Description
BUS_ID(integer)	Unique bus ID
AGENCY(string)	Agency that provided the record
DATE_AND_TIME(timestamp)	Date and time
ROUTE_ID(integer)	Unique bus route ID
LINE_ID(integer)	Unique bus line ID
RUN_ID(integer)	Unique run ID
ROUTE_DESCRIPTION(string)	Textual Description Of Route.
BUS_DIRECTION	NORTH,SOUTH,EAST,WEST,UNSPECIFIED
BUS_LOCATION	The bus location
BUS_LOCATION_TIME	The recorded timestamp
SCHEDULE_DEVIATION	The time deviation between the schedule and bus arrival time
NEXT_STOP_LOCATION	The location of the next stop
NEXT_STOP_TIME	The arrival time at the next stop
NEXT_STOP_SCHEDULED_TIME	The scheduled arrival time at the next stop
BRT_FLAG	The BRT status flag

TableName:CMS_INVENTORY

- Data: Static and dynamic information about changeable-message signs deployed on highways
- Metadata: Static information about the signs

Column	Description
DMS_ID(integer)	Unique Device id
AGENCY(string)	Agency That Provided The Record
DATE_AND_TIME (timestamp)	Date And Time
CITY(string;nullable)	City name
ON_STREET(string)	Street that ramp is on
FROM_STREET (string;nullable)	The beginning street name
TO_STREET (string;nullable)	The end street name
LOCATION(geometry)	Latitude And Longitude Of Sensor
DIRECTION(enum)	NORTH,SOUTH,EAST,WEST,UNSPECIFIED
POSTMILE(float)	Distance from a specific end of the road

TableName:CMS_DATA

- Data: Dynamic information about the state and content of each sign at each timestamp

Column	Description
DMS_ID(integer)	Unique device id
AGENCY(string)	Agency that provided the record
DATE_AND_TIME(timestamp)	Date and time
DEVICE_STATUS(string)	OK,FAILED
STATE(string)	DISPLAY,BLANK,NO_RESPONSE,UNKNOWN
DEVICE_TIME(timestamp)	The device time

TableName:EVENT_DATA

- Data: Special events, traffic incidents, and other unplanned events, including information like when the event happened or is planned to happen, what agencies were involved in the resolution, etc.

Column	Description
EVENT_ID(integer)	Unique event id
AGENCY(string)	Agency That Provided The Record
DATE_AND_TIME (timestamp)	Date And Time
ADMIN_CITY (string;nullable)	The administrative city
ON_STREET(string)	Street that ramp is on
FROM_STREET (string;nullable)	The beginning street name
TO_STREET(string;	The end street name

Data Access and Sharing

Although the dataset is collected and maintained by IMSC at our servers, all data in our traffic data repository are owned by LACMTA. We cannot release the data without approval from LACMTA. We plan to arrange approval from LACMTA for use of the data for research purposes at USC.