2020 25th International Conference on Pattern Recognition (ICPR)
Milan, Italy, Jan 10-15, 2021

Transfer Learning with Graph Neural Networks for Short-Term Highway Traffic Forecasting

Tanwi Mallick
Argonne National Laboratory
Lemont, Illinois
Email: tmallick@anl.gov

Prasanna Balaprakash
Argonne National Laboratory
Lemont, Illinois
Email: pbalapra@anl.gov

Eric Rask
Argonne National Laboratory
Lemont, Illinois
Email: erask@anl.gov

Jane Macfarlane
Lawrence Berkeley
National Laboratory
Berkeley, California
Email: jfmacfarlane@lbl.gov

Abstract—**Large-scale highway traffic forecasting approaches are critical for intelligent transportation systems. Recently, deep-learning-based traffic forecasting methods have emerged as promising approaches for a wide range of traffic forecasting tasks. These methods are specific to a given traffic network, however, and consequently they cannot be used for forecasting traffic on an unseen traffic network. Previous work has identified diffusion convolutional recurrent neural networks, (DCRNN), as a state-of-the-art method for highway traffic forecasting. It models the complex spatial and temporal dynamics of a highway network using a graph-based diffusion convolution operation within a recurrent neural network. Currently, DCRNN cannot perform transfer learning because it learns location-specific traffic patterns, which cannot be used for unseen regions of a network or new geographic locations. To that end, we develop `TL-DCRNN`, a new transfer learning approach for DCRNN, where a single model trained on a highway network can be used to forecast traffic on unseen highway networks. Given a traffic network with a large amount of traffic data, our approach consists of partitioning the traffic network into a number of subgraphs and using a new training scheme that utilizes subgraphs to marginalize the location-specific information, thus learning the traffic as a function of network connectivity and temporal patterns alone. The resulting trained model can be used to forecast traffic on unseen networks. We demonstrate that `TL-DCRNN` can learn from San Francisco regional traffic data and can forecast traffic on the Los Angeles region and vice versa.**

I. Introduction

With steadily increasing urbanization in major cities across the world, city planning and state-level department of transportation organizations are now focusing heavily on traffic management systems. The complexity of traffic management has, in the past, been too high a bar for most regional-level control-oriented technology solutions. Signal control, variable messaging signs, and traffic metering onto highways have managed at a localized level in the past and sufficed. With increased densities of vehicles on the road network, however, these solutions are sometimes falling short; and traffic is having a detrimental impact on area economics, productivity, emissions, and quality of life. As an example, the costs associated with congestion in California have been estimated to be on the order of $29B, with approximately $1.7B attributed to each of the Los Angeles (LA) and San Francisco (SFO) regions alone [1].

New data collection opportunities and advanced computing capabilities are beginning to emerge in transportation systems. Most of these are being deployed in a "corridor environment," where congested sections of major highways are instrumented to collect a variety of traffic measurements. These measurements are fed back to transportation management centers (TMCs) and decisions systems that, with the approval of humans in the loop, provide timely alerts to authorities and control traffic with cost measures that provide the best overall outcomes for the corridor, such as congestion metrics, fuel efficiency, and emissions. Beginning in the 1980s, TMCs have spread widely across the United States, with over 280 TMCs operating in the nation today. The decision systems in use by TMCs are a key component of an efficiently operating transportation system and the subject of significant research.

In parallel with the emerging data collection and computing capabilities afforded to TMCs, techniques to better forecast traffic conditions—a foundational component of a TMC's decision system—are also seeing significant improvements to the state of the art. Recently, deep learning methods such as deep belief networks [2] and stacked autoencoders [3], and recurrent neural network (RNN) variants [4] have emerged as promising approaches because of their ability to capture the long-term temporal dependencies. However, they do not model the spatial dependencies of highway network. Convolutional neural networks with recurrent units have been investigated to model the spatial temporal dynamics of traffic, where the spatial temporal traffic data are converted to a sequence of images and sequence-to-sequence learning is performed on the images [5]. These methods pose significant limitations, however, because they violate the fundamental non-Euclidean property of the network data. While the nearby pixels are correlated in images, nearby locations in a highway can be different because they can be on the opposite side of the highway (for example, one going into the city and one coming out of the city). To that end, Diffusion Convolutional Recurrent Neural Network (DCRNN) [6] has been proposed to overcome the challenges of spatiotemporal modeling of highway traffic networks. It is a state-of-the-art graph-based network that captures spatial correlation by a diffusion process on a graph and temporal dependencies using a sequence-to-sequence RNN.

The high performance of the deep learning models including DCRNN can be attributed to the availability of significant amounts of historical data for training. While these deep learning models can be transformational for the performance of a transportation decision system, many U.S. states do not have the long-standing data collection infrastructure that can provide such historical data for training, especially if these techniques are ultimately to be applied to both arterial roadways and highways. Moreover, the dynamic nature of transportation systems dictates that even for systems with highly instrumented corridors, other areas of emerging congestion may not have sufficient instrumentation or historical data. Relatedly, while several new data collection infrastructures have been deployed in various states, the time required to accumulate sufficient data for training can hinder model development and deployment. Furthermore, in the absence of an installed data collection infrastructure or areas where sensors are not as geospatially dense, probe data collected from GPS and cellphones can be used as a proxy to measure key traffic metrics such as speed. In these cases, model training can become difficult because historical data either may be unavailable or cannot be accumulated for privacy concerns. Furthermore, even if the historical data is available within a specific region, TMCs have limited computing capability and expertise to train deep learning models. These challenges, in the context of the improved traffic forecasting capabilities afforded by the DCRNN approach, suggest that identifying methods to deploy models trained in areas rich in historical data to areas with a paucity of data are especially promising. Additionally, if successful, these methods can allow states, cities, and municipalities to more quickly develop improved traffic forecasting capabilities with a significantly smaller infrastructure investment. Expanding the benefits from just TMCs, many other intelligent transportation applications, such as dynamic routing for freight traffic to congestion pricing based on forecast traffic conditions, could also benefit from localized models trained on data sets from more data-rich locations.

Transfer learning is a promising approach to circumvent possible data paucity, training, and deployment challenges. In this approach, a model trained for one task is reused and/or adapted for a related task. While transfer learning is widely used for image classification, sentiment analysis, and document classification in the text domain [7], it has received less attention in the traffic forecasting domain. Using transfer learning methods for graph-based highway traffic forecasting such as DCRNN is not a trivial task. The reason is that graphs have complex neighborhood correlations, as opposed to images, which have relatively simple local correlations since they are samples from the same Euclidean grid domain [5]. To address these issues, we have developed `TL-DCRNN`, a DCRNN with transfer learning capability. Given a large highway network with historical data, `TL-DCRNN` partitions the network into a number of regions. At each epoch, region-specific graphs and the corresponding time series data are used to train a single encoder-decoder model using minibatch stochastic gradient descent. Consequently, the location-specific traffic patterns are marginalized, and the model tries to learn the traffic dynamics as a function of graph connectivity and temporal pattern alone. We conduct extensive experiments on the real-world SFO and LA traffic dataset from the Performance Measurement System (PeMS) administered by the California Department of Transportation (Caltrans). Our contributions are as follows.

- We develop a new graph-partitioning-based transfer learning approach for DCRNN by marginalizing location-specific information to model the traffic dynamics as a function of temporal patterns and network connectivity.
- We demonstrate that our proposed transfer learning method can learn from SFO region data and forecast for the LA region and vice versa.

II. Problem setup

The short-term highway traffic forecasting problem can be defined on a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$, where $\mathcal{V}$ is a set of N nodes that represent highway sensor locations, $\mathcal{E}$ is the set of directed edges connecting these nodes, and $\mathcal{A} \in R^{N\times N}$ is the weighted adjacency matrix that represents the connectivity between the nodes in terms of highway network distance. The traffic state at time step t is represented as a graph signal $X_t \in R^{N\times F}$ on the graph $\mathcal{G}$, where F is the number of traffic metrics of interest (e.g., traffic flow, traffic speed, and density that change over time). Given H historical observations of the traffic state $X = (X_{t_1}, X_{t_2}, ..., X_{t_H}) \in \mathbb{R}^{H\times N\times F}$ and P observations of the current traffic state $X = (X_{t_1}, X_{t_2}, ..., X_{t_P}) \in \mathbb{R}^{P\times N\times F}$ on the graph $\mathcal{G}$, where $H >> P$, the goal is to develop a model that can forecast the traffic state of the next Q time steps on all nodes of the graph, $\hat{Y} = (\hat{X}_{t_{P+1}}, \hat{X}_{t_{P+2}}, ..., \hat{X}_{t_{P+Q}}) \in \mathbb{R}^{Q\times N\times F}$.

Let $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \mathcal{A}')$ be the graph with N' nodes that represents the highway network for which we do not have the historical time series data. Given P observations of the current traffic state $X' = (X'_{t_1}, X'_{t_2}, ..., X'_{t_P}) \in \mathbb{R}^{P\times N'\times F}$ on the graph $\mathcal{G}'$, the goal is to develop a model that can forecast the traffic state of the next Q time steps on all nodes of the graph $\mathcal{G}'$, $\hat{Y}' = (\hat{X}'_{t_{P+1}}, \hat{X}'_{t_{P+2}}, ..., \hat{X}'_{t_{P+Q}}) \in \mathbb{R}^{Q\times N'\times F}$.

In the context of Pan and Yang's transfer learning classification [8], our problem setup corresponds to the transductive transfer learning setting, where the source and target tasks are the same (short-term traffic forecasting on graphs), while the source and target domains (unseen regions of the highway network) are different but related.

III. Diffusion Convolution Recurrent Neural Network (DCRNN)

DCRNN is a state-of-the-art method for short-term traffic forecasting [6]. It is an encoder-decoder neural network architecture that performs sequence-to-sequence learning to carry out multistep traffic state forecasting. A simple and powerful variant of recurrent neural networks, called gated recurrent units (GRUs) [4], is used to design the encoder-decoder architecture. The matrix multiplications in GRUs is replaced with a diffusion convolution operation to make the DCRNN cell. In an L layered DCRNN architecture, each layer consists of R number of DCRNN cells. The DCRNN cell is defined by the following set of equations:

$$\begin{aligned} r^t &= \sigma(W_{r\bigstar\mathcal{G}}[X_t, h_{t-1}] + b_r) \\ u^t &= \sigma(W_{u\bigstar\mathcal{G}}[X_t, h_{t-1}] + b_u) \\ c^t &= tanh(W_{c\bigstar\mathcal{G}}[X_t(r_t \odot h_{t-1})] + b_c) \\ h_t &= u_t \odot h_{t-1} + (1 - u_t) \odot c_t, \end{aligned}$$

where X_t and h_t denote the input and final state at time t, respectively; r_t, u_t, and c_t are the reset gate, update gate, and cell state at time t, respectively; $\bigstar G$ denotes the diffusion convolution; and $W_r, W_u, andW_c$ are parameters for the corresponding filters. The diffusion convolution $\bigstar G$ operation over the input graph signal X and convolution filter W, which learns the representations for graph-structured data during training, is defined as

$$W_{\bigstar\mathcal{G}}X = \sum_{d=0}^{K-1} (W_O(D_O^{-1}\mathcal{A})^d + W_I(D_I^{-1}\mathcal{A})^d)X, \quad (1)$$

where K is a maximum number of diffusion steps; $D_O^{-1}\mathcal{A}$ and $D_I^{-1}\mathcal{A}$ are transition matrix of the diffusion process and the reverse one, respectively; D_O and D_I are the in-degree and out-degree diagonal matrices, respectively; and W_O and W_I are the learnable filters for the bidirectional diffusion process. The in-degree and out-degree diagonal matrices provide the capability to capture the effect of the upstream as well as the downstream traffic. The driving distances between sensor locations are used to build the adjacency matrix $\mathcal{A}$; a Gaussian kernel and a threshold τ parameter are used to sparsify $\mathcal{A}$.

During the training of DCRNN, a minibatch of time series sequence, each of length P from historical time series data X, is given as an input to the encoder. The decoder receives a fixed-length hidden representation of the data from the encoder and forecasts the next Q time steps for each sequence in the minibatch. The layers of DCRNN are trained by using backpropagation through time. DCRNN learns the weight matrices in Equation 1 by minimizing the mean absolute error (MAE) as a loss function.

IV. Transfer learning DCRNN

Similar to the convolution operation on images, the diffusion convolution $\bigstar G$ on a graph is designed to capture patterns that are local to a given node. This operation learns the latent representation of a diffusion process that starts from a given node to the neighboring connected node, and it is particularly suitable to capture the local diffusion behavior of traffic dynamics. From Equation 1, we can see that DCRNN becomes location-specific because of the presence of the weighted adjacency matrix $\mathcal{A}$ in the diffusion step. As a result, the convolution filter W is dependent on the given $\mathcal{A}$, which is kept constant throughout the training process. Our hypothesis is that if we change the graph $\mathcal{A}$ and the corresponding time series data during the training process, then we can make diffusion convolution filters generic as opposed to location-specific. Consequently, the resulting model becomes more generalizable and can be used to forecast traffic on unseen graphs.

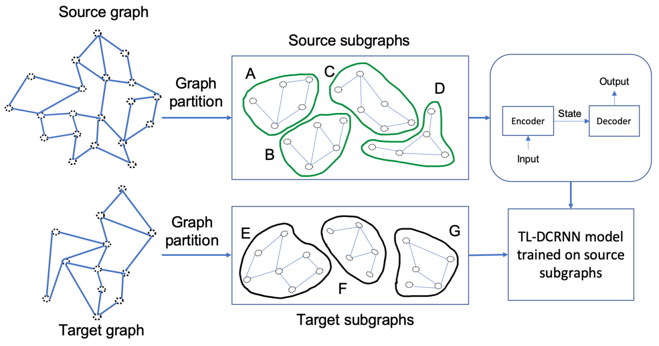

Fig. 1: TL−DCRNN partitions the large highway network with historical data into a number of subgraphs using a graph partitioning method. These subgraphs are used to train the the encoder-decoder architecture with diffusion convolution recurrent neural network cells. Given an unseen graph during inference, TL−DCRNN partitions the graph and uses the trained model for short-term traffic forecasting.

A high-level overview of the proposed TL−DCRNN is shown in Figure 1. Given the source graph $\mathcal{G}$ with the historical data X, TL−DCRNN first partitions it into a m subgraphs with equal numbers of n nodes using a graph partitioning method that takes the weighted adjacency matrix $\mathcal{A}$ of $\mathcal{G}$ as input. Let $\mathcal{G}_p = \{\mathcal{G}_p^1, \ldots, \mathcal{G}_p^m\} = \{(\mathcal{V}_p^1, \mathcal{E}_p^1), \ldots, (\mathcal{V}_p^m, \mathcal{E}_p^m)\}$, and $X_p = \{X_p^1, \ldots, X_p^m\}$ be the set of p subgraphs and their corresponding time series data. The minibatch stochastic gradient update of TL−DCRNN is the same as that of DCRNN, where for a given $\mathcal{G}_p^i$, a batch of input and output time series sequences from X_p^i is used to compute the errors and update the weights of the encoder and decoder architecture by backpropagation through time. The minibatch is constructed to preserve time ordering, a common approach in sequence-to-sequence time series modeling. The subgraph epoch for a given subgraph $\mathcal{G}_p^i$ uses all the data in X_p^i as a series of minibatches to update the parameters of the encoder decoder architecture. The epoch runs m subgraph epochs for each subgraph $\mathcal{G}_p^i \in \mathcal{G}_p$.

For inference, TL−DCRNN partitions the unseen target graph $\mathcal{G}'$ into m' subgraphs $\{\mathcal{G}'^1_p, \ldots, \mathcal{G}'^{m'}_p\}$ such that each subgraph $\mathcal{G}'^j_p$ has n nodes using the graph partitioning method. Given the current state of the traffic as a sequence of P time series on a subgraph $\mathcal{G}'^j_p$, the TL−DCRNN trained model forecasts the traffic for the next Q time steps.

From the learning perspective, the key difference between DCRNN and TL−DCRNN is that the former learns the location-specific spatiotemporal patterns on a static graph whereas the latter marginalizes the location-specific information and learns the spatiotemporal patterns across multiple subgraphs. Consequently, the model weights are trained in such way that it can provide generalization to similar but unseen graphs, a capability that can be used for forecasting on a unseen highway network.

Note on graph partitioning: Several graph partitioning methods exist that can be leveraged for partitioning the graphs in the training and inference phase. Typically, these methods cannot partition the graph into equal-sized subgraphs, but they can give partitions of similar sizes. In these cases, we can find the largest size of the graph and perform zero padding for all other subgraphs. The unseen graph $\mathcal{G}'$ does not need to be as large as $\mathcal{G}$. For a given $\mathcal{G}'$, when N' modulo n is relatively smaller than n, one can use zero padding for the subgraph whose size is smaller than n. When the modulo value is much smaller than n, the method of subgraphs with overlapping nodes [9] can be adopted, where a subgraph includes nodes from neighboring geographically close subgraphs.

Assumptions and implications: We assume that the graph $\mathcal{G}$ represents a large highway network such that it is amenable to graph partitioning and, in particular, the partitioned subgraphs expose a range of traffic dynamics. Otherwise, the transfer learning ability of the method will suffer. Furthermore, the transductive transfer learning assumptions of related domain discussed in [8] apply to our method. The highway traffic data can be seen as samples generated from a high-dimensional distribution parameterized by highway vehicle composition, infrastructure, and entry exit dynamics, among others. Given a completely different distribution sample, the model will not be able to perform transfer learning with high accuracy. For example, a model that is trained on certain regions of California will not generalize to highway network traffic forecasting in completely different regions of the world such as China and India, where the highway vehicle composition, infrastructure, and traffic dynamics can be dramatically different.

V. Experimental results

For the experimental evaluation, we used Cooley, a GPU-based cluster at the Argonne Leadership Computing Facility. It has 126 compute nodes, each node consisting of two 2.4 GHz Intel Haswell E5-2620 v3 processors (6 cores per CPU, 12 cores total), one NVIDIA Tesla K80 (two GPUs per node), 384 GB of RAM per node, and 24 GB GPU RAM per node (12 GB per GPU). The nodes are interconnected via an InfiniBand fabric. The software stack comprises Python 3.6.0, TensorFlow 1.3.1, NumPy 1.16.3, Pandas 0.19.2, and HDF5 1.8.17.

To generate subgraphs for the `TL-DCRNN` training, we use the multilevel k-way partitioning algorithm from Metis 5.1.0 [10]. This algorithm creates roughly k equal-sized partitions. Because of the fixed dimension of the input and the adjacency matrix used in the `TL-DCRNN`, all subgraphs should contain an equal number of nodes. We added rows and columns filled with zeros to make all the inputs and the adjacency matrices exactly equal in size.

We used the same set of hyperparameter values for all the training. These hyperparameter values were obtained from the open-source DCRNN implementation [11]: batch size, 64; filter type, random walk; maximum diffusion steps, 2; number of RNN layers, 2; number of RNN units per layers, 16; threshold `max_grad_norm` to clip the gradient norm to avoid exploring gradient problem of RNN, 5; initial learning rate, 0.01; and learning rate decay, 0.1. We used speed as the traffic forecasting metric. For training and inference, the forecast horizons were set to 60 minutes: the encoder gets 60 minutes (12 observations, one for every five minutes) of traffic data (time and speed), and the decoder outputs the forecasts for the next 60 minutes (12 predictions, one for every five minutes). Mean absolute error (MAE) was used as the training loss and test accuracy metric for comparing different methods.

TABLE I: Datasets used for the experiments. The source set (`src`) contains subgraphs from LA (or SFO), and the target set (`tgt`) contains subgraphs from SFO (or LA). The timelines of training, validation, and testing are given in the description.

Dataset	**Description**
Source subgraphs of LA (or SFO)	
`train-src`	≈36 weeks of time series data (1 Jan. 2018 to 13 Sept. 2018)
`val-src`	≈5 weeks of time series data (13 Sept. 2018 to 20 Oct. 2018)
`test-src`	≈10 weeks of time series data (20 Oct. 2018 to 31 Dec. 2018)
target subgraphs of SFO (or LA)	
`train-tgt`	≈36 weeks of time series data (1 Jan. 2018 to 13 Sept. 2018)
`val-tgt`	≈5 weeks of time series data (13 Sept. 2018 to 20 Oct. 2018)
`test-tgt`	≈10 weeks of time series data (20 Oct. 2018 to 31 Dec. 2018)

To compare the MAE values obtained from two models $M1$ and $M2$, we used a paired one-sided Wilcoxon signed-rank test with null hypothesis that the median difference between the two MAE distributions is greater than or equal to zero (MAE value from $M2$ is lower than or similar to that of $M1$). We compute the $p-$value from the test and reject the null hypothesis when the p-value is less than 0.05 (at a confidence level of 5%) in favor of the alternative that the median is less than zero (MAE value from $M1$ is lower than those of $M2$).

The source code for `TL-DCRNN` along with the dataset and the scripts to reproduce the experiments and results are available at https://github.com/tanwimallick/TL-DCRNN.

A. Transfer learning between LA and SFO regions

Here, we compare `TL-DCRNN` with DCRNN (without transfer learning) and show that the proposed transfer learning approach is effective.

We used the PeMS dataset [12] for the LA and SFO regions. It has 2,716 and 2,382 traffic sensor locations in LA and SFO for the entire year of 2018, respectively. Besides the time series data, PeMS captures spatial information such as the latitude and longitude of each sensor location. We computed the pairwise driving distances between the sensor locations using the latitude and longitude and built the adjacency matrix using a thresholded Gaussian kernel [6]. From one year of the time series data, we used 70% of the data (≈36 weeks) for training, 10% (≈5 weeks) for validation, and 20% (≈10 weeks) for testing. Consequently, we have six datasets. See Table I for the dataset summary and nomenclature adopted.

For `TL-DCRNN`, we partitioned the highway traffic network of LA and SFO into 15 and 13 subgraphs, respectively, to

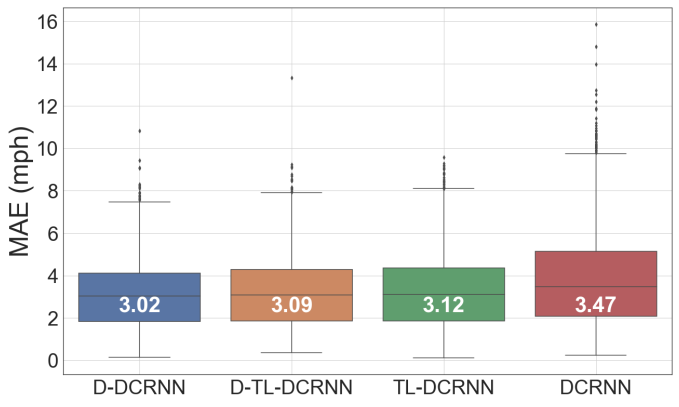

Fig. 2: Distributions of MAE values obtained by D-DCRNN, D-TL-DCRNN, TL-DCRNN, and DCRNN on the LA region.

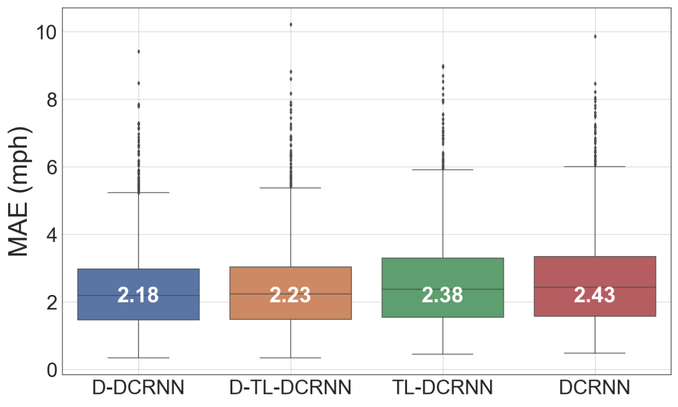

Fig. 4: Distributions of MAE of D-DCRNN, D-TL-DCRNN, TL-DCRNN, and DCRNN on the SFO region.

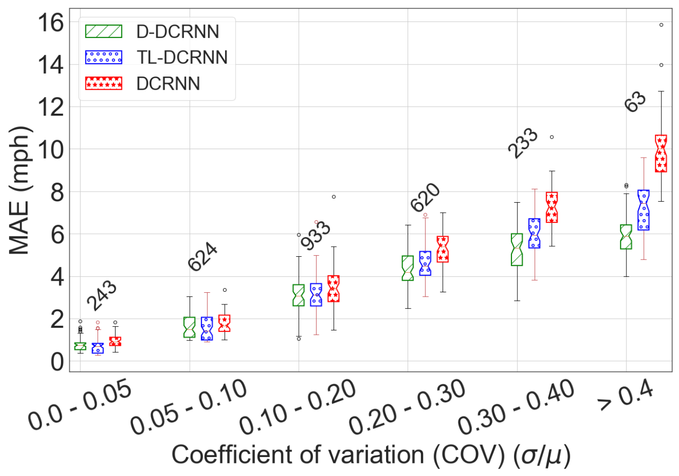

Fig. 3: Distributions of MAE of D-DCRNN, TL-DCRNN, and DCRNN with respect to coefficient of variation on the LA region. The numbers above show the number of observations (nodes) within the COV interval.

keep number of nodes per partition ($\approx$ 180) almost equal across the two datasets. We trained the TL-DCRNN models on train-src of LA (SFO) and tested them on test-tgt of SFO (LA), respectively. We compared our approach to DCRNN without transfer learning, where we trained it on the SFO (LA) dataset and applied it directly to forecast on the LA (SFO) dataset. Because of the fixed dimension of the input and adjacency matrix, we added rows and columns filled with zeros in the SFO dataset to make the inputs and the adjacency matrices equal in both the datasets. Moreover, as a best-case scenario, we included the DCRNN and TL-DCRNN model trained on SFO (LA) to forecast the traffic on SFO (LA). We call these models D-DCRNN and D-TL-DCRNN, where D stands for direct learning. Note that both DCRNN and D-DCRNN operate on the whole graph (without subgraphs).

Figure 2 shows the distributions of MAE values of the D-DCRNN, D-TL-DCRNN, TL-DCRNN, and DCRNN models on the LA dataset. We can observe that TL-DCRNN outperforms DCRNN. We found that the MAE values obtained by TL-DCRNN are lower than those obtained by DCRNN on 2,497 out of 2,717 nodes. The observed differences were significant according to a paired one-sided Wilcoxon signed-rank test, which showed a $p-$value of 0.00 for the TL-DCRNN and DCRNN comparison. To get further insight, we plot the MAE values with respect to the coefficient of variation (COV). This is given by the ratio of the standard deviation and the mean of the time series for each node in test-tgt. This metric can be used as a proxy to measure the traffic dynamics: smaller values indicate that the speed is stable (less dynamic), and larger values mean that a wide range of speed values has been observed (more dynamic). The plot of the MAE values with respect to COV shows how forecasting error changes with the increasing traffic dynamics. We binned the COV into 5 buckets 0.0 to 0.05, 0.05 to 0.10, 0.10 to 0.20, 0.20 to 0.30, 0.30 to 0.40 and 0.40 and above. The results are shown in Figure 3. We can observe that the differences between TL-DCRNN and DCRNN increase as a function of COV. The observed differences are rather small when COV values are less than 0.2; however, they become significant as the COV values become larger.

Figure 4 shows the results obtained on SFO. We can observe a similar trend. The median of MAE values are 2.18, 2.23, 2.38, and 2.43 for D-DCRNN, D-TL-DCRNN, TL-DCRNN, and DCRNN models, respectively. We found that the MAE values obtained by TL-DCRNN are lower than those obtained by DCRNN on 1,495 out of 2,383 nodes. The observed differences are significant according to a paired one-sided Wilcoxon signed-rank test, which showed a $p-$value of 5.02×10^{-68} for TL-DCRNN and DCRNN comparison. Figure 5 shows the distributions of MAE as a function of COV intervals. The distributions show that MAE values obtained by TL-DCRNN are lower than DCRNN in 4 out of the 6 COV intervals. The number of nodes with COV larger than 0.3 is small (36) in SFO dataset. Hence, the differences between the three methods were not significant.

Figure 6 shows the distribution of pairwise MAE differences between TL-DCRNN vs D-DCRNN (direct learning, best case)

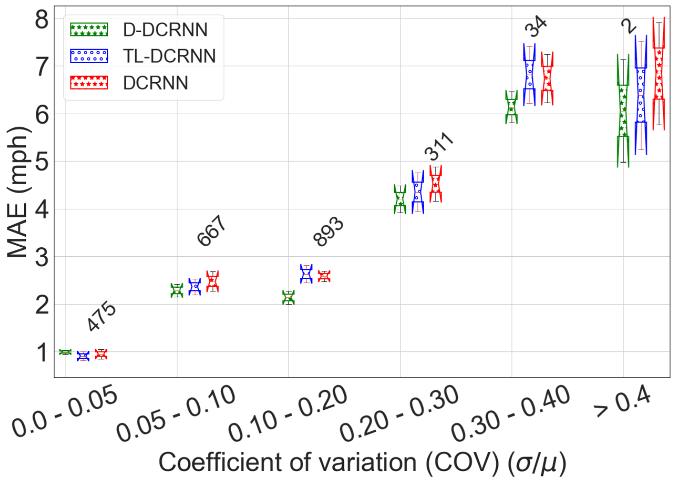

Fig. 5: Distributions of MAE of TL−DCRNN, DCRNN, and D-DCRNN with respect to coefficient of variation on the SFO region. The numbers above each box show the number of observations (nodes) in the distribution.

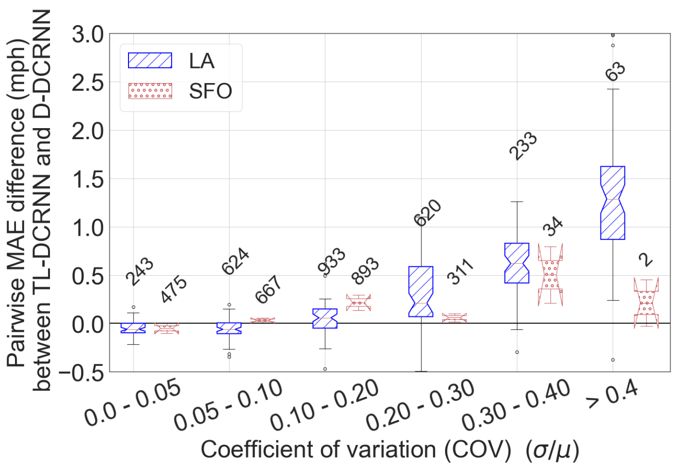

Fig. 6: Pairwise MAE differences between TL−DCRNN and D-DCRNN and with respect to COV values computed on test-tgt on LA and SFO regions. The numbers above show the number of observations (nodes) within the COV interval.

as a function of COV intervals for SFO and LA datasets. The MAE values obtained by D-DCRNN are not significantly better than that of TL−DCRNN for less dynamic traffic sensors. On LA and SFO datasets, the two methods achieve comparable results up to COV values of 0.30 and 0.10 (also see Figures 3 and 5). These results imply that when the traffic is less dynamic, transfer learning will be as good as the direct learning. On LA data, we can see a clear trend in which an increase in the COV values increases the MAE difference values and D-DCRNN forecasts become more accurate than that of TL−DCRNN. This is because D-DCRNN models trained and tested on the same graph captures the location-specific traffic dynamics better than TL−DCRNN that was trained on the SFO dataset and tested on LA. We can observe a similar trend in the SFO dataset as well. These results show that for locations with more traffic dynamics, learning the location-specific spatial-temporal patterns is critical for higher accuracy.

In SFO dataset, the significant drop in the differences for COV larger than 0.4 can be attributed to the small number (2) of nodes. Moreover, the number of nodes with COV larger than 0.3 is rather small (36) when compared with the LA dataset. This can be attributed to the fact that SFO traffic is less dynamic when compared with LA traffic.

TABLE II: Accuracy metrics comparison of TL-DCRNN with other short-term traffic forecasting approaches.

Method	MAE	RMSE	MAPE
Training and testing on PEMS-BAY			
ARIMA [13]	3.38	6.50	8.30%
SVR [14]	3.28	7.08	8.00%
FNN [15]	2.46	4.98	5.89%
FC-LSTM [4]	2.37	4.96	5.70%
STGCN [16]	2.49	5.69	5.79%
DCRNN [6]	2.07	4.74	4.90%
GMAN [17]	1.86	4.32	4.31%
Training on LA and testing on PEMS-BAY			
TL-DCRNN	2.13 ± 1.09	5.23 ± 2.29	5.55 ± 4.34

B. Comparison with other methods

Here, we present our comparison of TL−DCRNN with other state-of-the-art short-term traffic forecasting methods under two different settings on a commonly used PEMS-BAY benchmark dataset and demonstrate the effectiveness of TL−DCRNN.

1) TL−DCRNN vs direct learning: We compared TL−DCRNN with a number of methods proposed in the literature: (1) autoregressive integrated moving average (ARIMA) [13], which considers only the temporal relationship of the data; (2) support vector regression (SVR) [14], a linear SVR for forecasting; (3) a feed-forward neural network (FNN) [15] with two hidden layers; (4) fully connected LSTM (FC-LSTM) [4] with encoder-decoder architecture; (5) a spatiotemporal graph convolutional network (STGCN) [16], which combines graph convolutions and gated temporal convolutions; (6) a diffusion convolutional recurrent neural network (DCRNN) [6], as discussed in Section III; and (7) a graph multiattention network (GMAN) [17], an encoder-decoder architecture with multiple spatiotemporal attention. All the methods were trained and tested on the PEMS-BAY dataset, which is a widely used benchmark for short-term traffic forecasting methods. This dataset has 325 sensors in the Bay Area with 6 months of time series data ranging from Jan. 1, 2017, to June 30, 2017. We used 70% of the data for training (Jan. 1, 2017, to May 7, 2017), 10% of the data for validation (from May 7, 2017, to May 25, 2017), and 20% of the data for testing (from May 25, 2017, to June 30, 2017). We used the accuracy results reported in [17], where GMAN results are compared with other methods. The TL−DCRNN model is *trained on LA dataset and tested on the PEMS-BAY dataset*. We used the same timeline for training, validation, and testing as the PEMS-BAY dataset.

In addition to MAE, we used root mean square error (RMSE) and mean absolute percentage error (MAPE) metrics

TABLE III: Transfer learning performance comparison of TL-DCRNN and other traffic forecasting methods. TL-DCRNN achieves the best performance with all three metrics.

Models	MAE	RMSE	MAPE
STGCN [16]	6.53 ± 2.69	10.07 ± 3.47	13.31 ± 6.38 %
FC-LSTM [4]	4.69 ± 1.79	8.48 ± 3.17	12.32 ± 8.78 %
GMAN [17]	4.05 ± 1.56	7.57 ± 2.51	8.5 ± 4.58 %
DCRNN [6]	3.3 ± 1.24	6.91 ± 2.19	8.21 ± 5.57 %
TL-DCRNN	2.13 ± 1.09	5.23 ± 2.29	5.55 ± 4.34 %

to compare the accuracy of the models. The results are shown in Table II. Although TL-DCRNN is trained on LA dataset, it achieves MAE (2.13 ± 1.09), RMSE (5.23 ± 2.29), and MAPE (5.55 ± 4.34) distribution values that are superior to ARIMA, SVR, FNN, FC-LSTM, and STGCN. Despite the fact that DCRNN and GMAN were trained on the PEMS-BAY dataset, their accuracy metrics were not significantly better than those obtained by TL-DCRNN trained on the LA dataset.

2) Comparison between transfer learning models: We trained STGCN, FC-LSTM, GMAN, DCRNN, and TL-DCRNN on the LA dataset and tested on the PEMS-BAY dataset. These two datasets have 6 months of time series data, and we used the same timeline as described in V-B1. The models were trained and validated on 70% and 10% of the LA dataset; 20% PEMS-BAY dataset was used for evaluation. The LA dataset has 2,683 sensors location, and the PEMS-BAY dataset has 325 sensors location. We partitioned the LA highway traffic network into 9 partitions and used zero padding to make all the adjacency matrices of the size 325. For STGCN, FC-LSTM, GMAN, and DCRNN, we trained 9 partition-specific models for each method. These models were evaluated on the validation dataset. For each method, we selected the model with the lowest validation MAE and evaluated on the PEMS-BAY dataset. For TL-DCRNN, we trained a single model that utilizes all 9 subgraphs of LA dataset and evaluated it on the PEMS-BAY dataset. We did not include ARIMA, SVR, and FNN for the comparison because they can learn individual time series data but are unable to learn the entire network.

The results are shown in Table III. We observe that TL-DCRNN achieves MAE, RMSE, and MAPE distribution values, which are better than STGCN, FC-LSTM, GMAN, and DCRNN. The observed differences were significant according to a paired one-sided Wilcoxon signed-rank test, which showed $p-$value of 0.00 for for all the pairwise comparisons. The MAE values obtained by TL-DCRNN are lower than those obtained by STGCN, FC-LSTM, GMAN, and DCRNN on 323, 320, 321, and 304 out of 325 nodes, respectively. Similarly, the RMSE values obtained by TL-DCRNN are lower than those obtained by STGCN, FC-LSTM, GMAN, and DCRNN on 310, 295, 313, and 270 out of 325 nodes, respectively. Moreover, the MAPE values obtained by TL-DCRNN are lower than those obtained by STGCN, FC-LSTM, GMAN, and DCRNN on 308, 302, 300, and 294 out of 325 nodes, respectively.

VI. Related work

Graph-convolution-based forecasting models have shown a significant improvement in traffic forecasting tasks over classical approaches such as ARIMA and Kalman filtering, which are not effective in capturing complex spatial and temporal correlations [13]. Cui et al. [18] developed a graph convolutional long short-term memory network. They used the graph convolution operation inside the LSTM cell with regularization approaches. Yu et al. [16] integrated graph convolution and gated temporal convolution in a spatiotemporal convolutional block for traffic forecasting. Li et al. [6] proposed a DCRNN method that models traffic state as a diffusion process on a graph and used it within a GRU cell. All these methods, however, cannot perform forecasting on unseen graphs because they learn location-specific traffic dynamics and require the same highway network for training and inference.

The prior work on transfer learning for short-term traffic forecasting is sparse. Wang et al. [19] proposed an image-based convolutional LSTM network to perform transfer learning for crowd flow prediction from a data-rich city to a data-scarce city. The method first learns a matching function using Pearson correlation to find a similar source city for each target city. During training of the network the method tries to minimize the hidden representations of the target region and its matched source region inside the loss function. This approach does not incorporate multiple nodes, however, and does not take into account the spatial graph dependency. Recently, Yao et al. [20] proposed a metalearning method for traffic volume and water quality prediction. This approach captures knowledge from multiple nodes. It uses an image-based convolutional LSTM network to train on multiple source nodes and uses those trained weights for prediction on the target nodes. The method uses spatial-temporal memory to store representation of diffident regions of the source cities. The regions are found by $k-$means clustering on the averaged 24-hour patterns of each region, and region-specific weights stored in the memory are utilized for prediction via an attention mechanism. Krishnakumari et al. [21] developed a method that first clusters the feature vectors obtained from the pretrained image-based convolutional network and then uses the cluster to predict one-step forecast for the similar target location using an ensemble of multiple models such as multilayer perceptron, random forest, K-nearest neighbor, support vector machine (SVM), and Gaussian process. Xu1 et al. [22] and Lin [23] conducted preliminary studies for traffic prediction using cross city transfer leaning using SVM and dynamic time warping, respectively. Fouladgar et al. [24] proposed a transfer learning method using an image-based convolutional network and LSTM for traffic forecasting in case of congestion. None of these methods use graph convolution to model the spatial dependencies, and they cannot be applied directly to short-term highway forecasting.

TL-DCRNN is inspired by the cluster-GCN [25] training, where a graph convolution network training is proposed for learning tasks on large graph classification problems. In this approach, each batch for stochastic-gradient-descent-based training uses samples of subgraphs of the original graph. Cluster-GCN is built for node and link classification on graphs, however, and it cannot be used to model graph diffusion and temporal characteristics of the traffic data.

VII. Conclusion and Future Work

We developed `TL-DCRNN`, a graph-partitioning-based transfer learning approach for a diffusion convolution recurrent neural network to forecast short-term traffic on a highway network. `TL-DCRNN` partitions the data-rich source highway network into a number of regions and learns the spatiotemporal traffic dynamics as a function of the traffic state and the network connectivity by marginalizing the location-specific patterns. The trained model from `TL-DCRNN` is then used to forecast traffic on unseen regions of the highway network. We demonstrated the efficacy of `TL-DCRNN` by showing that the approach can perform transfer learning between LA and SFO regions. `TL-DCRNN` outperformed a number of methods developed for traffic forecasting despite being applied to a region unseen in training, whereas the other methods were both trained and applied on the same region. Moreover, `TL-DCRNN` outperformed all state-of-the-art traffic forecasting methods in a transfer learning setting. Allowing practitioners to apply data-driven methods trained on datasets collected elsewhere is a transformative capability, enabling a wide range of transportation system operations and functions to operate more efficiently and sustainably through improved forecasting at reduced infrastructure development and data acquisition costs.

Our future work will include (1) deployment strategies for traffic management systems, which can vary across the country; (2) transfer learning capability for alternate data sources such as mobile device data to relieve the cost of installing infrastructure sensors and to investigate performance for non-highway applications (i.e., arterials); (3) metalearning for graph-based transfer learning for highway networks; and (4) network structural implications for extending this approach beyond highway implementations, which may include characterizing how graph constraints are encoded in the DCRNN.

Acknowledgments

This material is based in part upon work supported by the U.S. Department of Energy, Office of Science, under contract DE-AC02-06CH11357. This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility under contract DE-AC02-06CH11357. This report and the work were sponsored by the U.S. Department of Energy (DOE) Vehicle Technologies Office (VTO) under the Big Data Solutions for Mobility Program, an initiative of the Energy Efficient Mobility Systems (EEMS) Program. David Anderson and Prasad Gupte, the DOE Office of Energy Efficiency and Renewable Energy (EERE) managers played important roles in establishing the project concept, advancing implementation, and providing ongoing guidance.

References

[1] TRIP, "California transportation by the numbers," August 2018. [Online]. Available: https://tripnet.org/wp-content/uploads/2018/08/CA_Transportation_by_the_Numbers_TRIP_Report_Aug_2018.pdf

[2] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: deep belief networks with multitask learning," *IEEE Trans. on Intel. Trans. Sys.*, vol. 15, no. 5, pp. 2191–2201, 2014.

[3] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Trans. on Intel. Trans. Sys.*, vol. 16, no. 2, pp. 865–873, 2015.

[4] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *2016 31st Youth Academic Annual Conf. of Chinese Assoc. of Automation (YAC)*. IEEE, 2016, pp. 324–328.

[5] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[6] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Intl. Conf. on Learning Representations (ICLR '18)*, 2018.

[7] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *arXiv preprint arXiv:1911.02685*, 2019.

[8] S. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009.

[9] T. M., P. B., E. R., and J. M., "Graph-partitioning-based diffusion convolution recurrent neural network for large-scale traffic forecasting," *arXiv preprint arXiv:1909.11197*, 2019.

[10] "METIS – serial graph partitioning and fill-reducing matrix ordering," http://glaros.dtc.umn.edu/gkhome/metis/metis/overview, 2016, accessed: 2020-03-19.

[11] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," https://github.com/liyaguang/DCRNN, 2018.

[12] "Caltrans performance measurement system (PeMS)," http://pems.dot.ca.gov/, 2019.

[13] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *Journal of Trans. Eng.*, vol. 129, no. 6, pp. 664–672, 2003.

[14] C. Wu, J. Ho, and D. Lee, "Travel-time prediction with support vector regression," *IEEE transactions on intelligent transportation systems*, vol. 5, no. 4, pp. 276–281, 2004.

[15] M. Raeesi, M. Mesgari, and P. Mahmoudi, "Traffic time series forecasting by feedforward neural network: a case study based on traffic data of Monroe," *The Intl. Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 40, no. 2, p. 219, 2014.

[16] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," *arXiv preprint arXiv:1709.04875*, 2017.

[17] C. Zheng, X. Fan, C. Wang, and J. Qi, "Gman: A graph multi-attention network for traffic prediction," *arXiv preprint arXiv:1911.08415*, 2019.

[18] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," *arXiv preprint arXiv:1802.07007*, 2018.

[19] L. Wang, X. Geng, X. Ma, F. Liu, and Q. Yang, "Cross-city transfer learning for deep spatio-temporal prediction," *arXiv preprint arXiv:1802.00386*, 2018.

[20] H. Yao, Y. Liu, Y. Wei, X. Tang, and Z. Li, "Learning from multiple cities: A meta-learning approach for spatial-temporal prediction," in *The World Wide Web Conference*. ACM, 2019, pp. 2181–2191.

[21] P. Krishnakumari, A. Perotti, V. Pinto, O. Cats, and H. van Lint, "Understanding network traffic states using transfer learning," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 1396–1401.

[22] F. F. Xu, B. Y. Lin, Q. Lu, Y. Huang, and K. Q. Zhu, "Cross-region traffic prediction for China on OpenStreetMap," in *Proceedings of the 9th ACM SIGSPATIAL International Workshop on Computational Transportation Science*. ACM, 2016, pp. 37–42.

[23] B. Y. Lin, F. F. Xu, E. Q. Liao, and K. Q. Zhu, "Transfer learning for traffic speed prediction: A preliminary study," in *Workshops at the Thirty-Second AAAI Conf. on Artificial Intelligence*, 2018.

[24] M. Fouladgar, M. Parchami, R. Elmasri, and A. Ghaderi, "Scalable deep traffic flow neural networks for urban traffic congestion prediction," in *2017 Intl. Joint Conf. on Neural Networks (IJCNN)*. IEEE, 2017, pp. 2251–2258.

[25] W. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C. Hsieh, "Cluster-GCN: An efficient algorithm for training deep and large graph convolutional networks," *arXiv preprint arXiv:1905.07953*, 2019.