

Neural Methods in Urban Data Science

An Yan and Bill Howe, University of Washington

1 Introduction

Today, more than 54 percent of the world's population is living in urban area [1]. Predicting dynamic urban activities such as energy consumption, air pollution, public safety, and traffic flows has become a fundamental task for improving the quality of human life. Urban mobility is closely intertwined with many important problems in urban areas: it is crucial to employment opportunities and access to resources such as education and health care [2]. Prediction tasks for urban mobility involves transport resource demand prediction, travel time prediction, traffic flows prediction, ride-hailing waiting time estimation, and driving behavior analysis for automatic driving cars, etc.

Predicting urban mobility events is challenging. One key challenge is to capture the temporal dynamics. For example, taxi demand exhibits temporal patterns such as daily, weekly, and seasonal cycles, and is influenced by factors such as weather, holidays, and public events. It is also challenging to capture dependencies in space. According to the first law of geography: everything is related to everything else, but near things are more related than distant things. For mobility events, neighboring regions often share similar properties or influence each other. In addition, the spatial context is important for predicting mobility events. For instance, ride-hailing demand in one region is affected by the density of Place of interests (POI) of it. Compared to modeling space dependencies, it is more challenging to model the dynamic interactions between space and time. For example, in traffic prediction, the relationship between a residential area and an employment center is strong in workday peak hours; whereas the relation between a residential area and a restaurant area becomes strong during evenings on weekends [3]. Another prominent challenge lies in data integration. As open urban data with various dimensions and forms becomes increasingly available, how to extract meaningful information from these heterogeneous data sources and integrate them into prediction model presents a challenge [4].

Deep learning [5] has achieved tremendous success in modeling sequence data [6] such as language and data with spatio-temporal context such as videos [7]. Applications of deep learning to urban mobility are in their infancy, but promising examples suggest that deep learning is well-positioned to address the challenges mentioned above (see Appendix). The rest of this paper is organized as follows. Section 2 briefly reviews conventional machine learning methods for spatio-temporal modeling. Section 3 reviews the state of the art deep learning methods, along with relevant data fusion techniques for spatio-temporal settings using urban mobility examples including one example of my own work. Advantages and limitations of each method are discussed. Section 4 discusses future research directions, and Section 5 summarizes the limitations of machine learning approaches for urban mobility studies.

2 Conventional Machine Learning for Spatio-temporal Modeling

Early work adopts time series analysis methods such as autoregressive integrated moving average (ARIMA) to predict urban mobility events [8, 9, 10]. This family of methods assumes input time series is stationary and relies on manual tuning to set seasonality and other parameters. They are not able to handle complex nonlinear spatial-temporal correlations. Conventional machine learning methods such as random forest (RF) and support vector machines (SVM) have been widely employed in urban settings [11]. For example, Li et al. [12] uses Gradient Boosting Regression Trees (GBRT) to predict the number of check-ins and check-outs of each station for bikeshare systems. Compared to time series analysis, machine learning methods have weaker assumptions of input data distribution and stronger predictive power. Nevertheless, conventional machine learning relies on hand-crafted features to capture spatio-temporal context. For example, taxi demand of the last hour and average demand of last week are used to capture the most recent status and longer history of the system, respectively. Hand-designed features are interpretable, but also tedious, ad-hoc, and

inflexible. Moreover, they rarely exploit spatio-temporal dependencies exhaustively as some latent factors are not directly observable [13].

3 Deep Learning for Spatio-temporal Modeling

Deep learning is a family of machine learning algorithms that uses artificial neural networks which typically contain many layers. Compared to conventional machine learning, deep learning has several advantages: 1) exceptional predictive power due to its ability to learn complex non-linear associations; 2) ability to learn spatial or temporal dependencies through specific network structures such as recurrent neural networks (RNN) [14] and convolutional neural networks (CNN) [5]; 3) automatic extraction of features from input data; and 4) end-to-end learning of powerful feature representations which are potentially reusable, shareable, and transferable among multiple tasks.

Two deep learning architectures: RNN and CNN, are particularly useful in modeling spatio-temporal dynamics in urban mobility. RNN and its variants can capture time dependencies. They are usually used for language modeling but can also be used for predicting time series of mobility events. CNN and its variants can model spatial structures. Originally, they are used for computer vision tasks such as image classification and object detection, but can also be adapted to predict citywide mobility events such as traffic flows (see Appendix). To capture spatial and temporal dependencies simultaneously in one network, network architectures such as ConvLSTM [15] and 3D CNN [7] have been developed. We refer this type of network as Spatio-temporal Neural Network (STNN) throughout this paper. STNNs were proposed primarily for video prediction, recently there has been efforts to adapt them to urban mobility [16, 17].

One prominent challenge is that urban data typically have multiple modalities (e.g., weather, traffic volume, etc) with different input dimensions which a single deep learning architecture alone can't solve. For example, weather data such as temperature and precipitation are usually time series (one dimensional); geospatial data such as road networks and POIs are two dimensional; and traffic volumes and collisions are three-dimensional. How to integrate them into one model for spatio-temporal (three dimensional) prediction raises challenges. To tackle these challenges, data fusion techniques are often employed along with deep learning for problems in urban settings.

This section first introduces data fusion techniques for deep learning, followed by descriptions of several classes of deep learning based methods for spatio-temporal predictions including RNN-based models, CNN-based models, and STNN models using urban mobility examples. Advantages and limitations of each deep learning architecture and each data fusion technique are discussed. An example of my own work is presented at the end to compare different methods for new mobility resource demand prediction.

3.1 Heterogeneous data fusion in deep learning

Data fusion is popular set of methods in deep learning to handle multi-modal data. Urban mobility is a typical problem with heterogeneous data. The use of exogenous urban information can 1) contribute to accurate prediction, and 2) provide spatial or temporal context. There are two major ways to perform data fusion in deep learning: feature-level fusion and model fusion [4].

Feature-level fusion. These methods directly fuse (i.e. concatenate) different data sources at the feature level into one feature vector (or feature map). Then the fused feature vector (or feature map) is fed into a single machine learning model. This is sometimes referred to as *early fusion* and is the most commonly used method for machine learning. Feature-level fusion treats each individual data source equally without considering their semantic meanings.

Model fusion. These methods use separate submodels to learn feature representations from different data sources simultaneously, and fuse (e.g., directly concatenate, element-wise product, etc.) the learned feature representations for further learning. This is sometimes referred to as *late fusion* [18]. Model fusion is based on the understanding of the relationship among different data sources. For instance, the algorithm designer has to know which features can be grouped together to be fed into one submodel. Compared to feature-level fusion, model fusion usually requires more complex module structures and introduces extra computation cost. However, it usually achieves better performance than feature-level fusion.

3.2 RNN-based ST models

RNNs have achieved numerous success in sequence-to-sequence prediction problems such as language translation, speech recognition, and question answering. The basic unit is a RNN node (or cell) that maintains an internal memory of the past and simultaneously takes a new input (e.g., a word in a sentence). These cells are connected to form an ordered sequence. In this way, when making a decision, RNN accounts for current input as well as previous history recorded in its memory. Long short-term memory (LSTM) is a type of RNN cell that is able to exploit long-term

temporal dependencies [14]. LSTM has been applied to many urban mobility problems [19]. For example, Xu et al. [20] used LSTM to predict dockless shared bike demand in Nanjing, China. Their experiments showed that LSTM outperforms several conventional machine learning methods including XGBoost, SVM, and ANN; as well as deep learning methods such as regular RNN. Yu et al. [21] applied LSTM to forecast peak-hour traffic. They also found that LSTM gives superior performance than linear regression, ridge regression, feed-forward network, and random walk. However, the above mentioned studies do not consider the spatial relations.

To incorporate spatial context into RNN, special treatments to either the network structure or features are needed. For example, ST-RNN [22] incorporated distance-specific transition matrices to capture spatial dependencies. Wang et al. [23] proposed a graph-structured RNN for modeling sparse spatio-temporal data. The basic idea is to construct a graph with each node representing a region of the city (not necessarily grid region), and an LSTM is imposed to fit the time series of that node. The LSTM of each node receives the outputs of neighboring nodes with weights learned from the Hawkes process. These two examples belong to the category of model structure modification. The following example shows in greater details how to use feature-level data fusion to capture spatial context in RNN.

Example: Traffic speed prediction with LSTM

Liao, et al. [24] proposed an LSTM based model for predicting future traffic speed $\{v_{t+1}, v_{t+2}, \dots, v_{t+t^0}\}$ given previous traffic speed $\{v_1, v_2, \dots, v_t\}$ for a road segment in a city. Specifically, traffic speed is sampled at 15-minute intervals and 24 hours of sequence ($t = 96$) is used to predict the next 2-hour ($t^0 = 8$) traffic speed. Traffic speed data is collected from 15,073 selected road segments in Beijing during April 2017 to May, 2017 from the Baidu Map application [1].

They first proposed a sequence-to-sequence model (Seq2Seq) for mere temporal pattern prediction from the past records without any exogenous features. The encoder extracts information from historical traffic speed sequence, and the decoder uses the extracted information to predict traffic speed of next time steps. Based on this Seq2Seq model, the

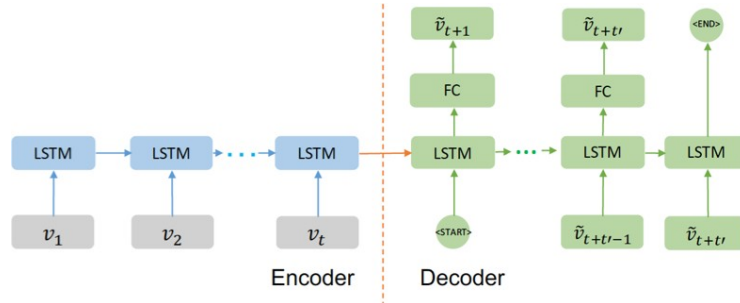


Figure 1: Seq2Seq: The Sequence to Sequence model predicts future traffic speed $\{v_{t+1}, v_{t+2}, \dots, v_{t+t^0}\}$, given the previous traffic speed $\{v_1, v_2, \dots, v_t\}$. [24]

authors employed three fusion strategies to incorporate geographical and social attributes, spatial context, and online map query impacts into the model.

- Feature-level fusion with raw features. The features include characteristics of each road segment such as number of lanes and speed limit, and social attributes such as public holidays, peak hours, and workdays. These features are concatenated directly into decoder part of the model. This procedure is similar to feature construction in conventional machine learning approaches.
- Feature-level fusion with learned embeddings (representation). To incorporate the spatial dependencies, the authors embedded the traffic speed of neighboring road segments using graph convolutional neural network (graph CNN) [25] and concatenated the learned embeddings to the encoder network. This procedure seeks to construct a feature that captures spatial context. Instead of hand-code the feature, the authors used graph CNN to learn the feature.
- Model fusion. Lastly, the authors found out a correlation between online map query counts and traffic speed. They calculated a time sequence of query impact (QI) based on query counts, and used another LSTM encoder whose final hidden states were concatenated into the decoder network. This procedure fuses the output of two submodels (two encoders) to form a new representation to be used in subsequent submodels (i.e. decoder).

<https://map.baidu.com/>

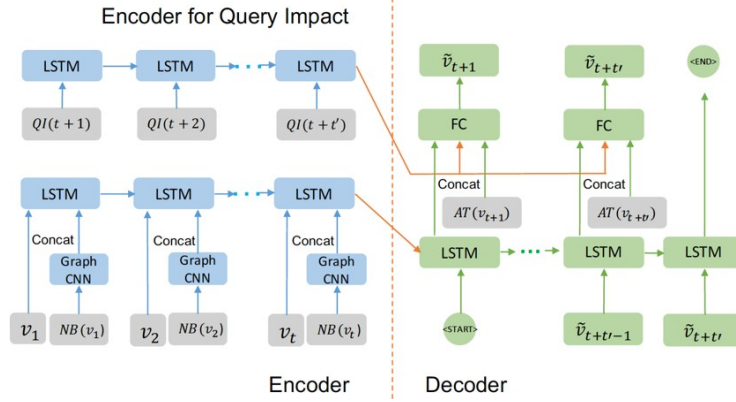


Figure 2: An LSTM-based model that integrates exogenous information including attributes $AT(v_t)$, spatial relation $NB(v_t)$, and query impact $QI(1, t)$ [24]

They evaluated this model with mean absolute percentage error (MAPE) and found out that it outperformed two conventional machine learning algorithms: random forest (RF) and support vector regression (SVR).

This example shows how LSTM can be applied to predict time series at different locations (discrete road segments) of a city. Because there are spatial relations among different road segments, treating them independently is sub-optimal. The authors used feature-level data fusion to incorporate a feature that encodes spatial relations.

Strengths of RNN-based ST models

- They are capable of modeling complex temporal dependencies without explicitly using temporal features.
- They are suited for spatially discrete data or data that is sparse in space and time. For example, predicting traffic speed of sparse road segments or predicting bikeshare demand for each station. In these cases, we only concern about certain fixed regions or spots in a city.

Limitations of RNN-based ST models

- They typically cannot capture very long time dependencies. As the input sequence gets longer (e.g., more than 300), the ability of LSTM to capture signals from past gets weaker [26].
- They need extra treatment to model structure or features to capture spatial dependencies.

3.3 CNN-based ST models

CNN has gained enormous popularity in computer vision tasks and has been deployed in numerous real-world applications to perform face recognition, image retrieval, and medical image analysis, etc. CNN can capture local spatial context by convolutional operations (or filters). A 2D convolutional filter is a weighting matrix that slides across the image and extracts increasingly sophisticated features from edges to semantically meaningful objects as layers progress deeper [13]. Broader spatial context can be learned through pooling operations.

CNN usually requires lattice like data as input (graph CNN is an exception). One way to transform spatio-temporal urban data into image-like form is to use space-time matrix. For example, Ma et al. [27] uses a time-space matrix to represent network-wide traffic speed. The columns of the matrix represent different road sections and the rows represent different time intervals. Although this time-space matrix can record spatial and temporal information at the same time, it squeezes 2D space into 1D representation, thus much spatial context is lost.

A more common way to represent urban data for CNN is to partition the study area into equal-sized square grids. Each grid resembles a pixel of an image. For examples, taxi requests data can be transformed into 2D matrix by counting the number of requests in each grid. This method is natural for spatially continuous data but not for discrete, sparse, or network-like data. For example, in traffic speed prediction, spatial correlation is dominated by road network structure, not proximity. Consider the case where two roads locate in the opposite direction of the highway, their traffic speed may differ significantly though they are physically close to each other [28]. The following example used a grid partition method to represent urban data. Temporal context is integrated into CNN-based models through model fusion, that is, training separate submodels using CNN for different time period and optimizing them jointly.

Example: Predicting Citywide Crowd Flows

Zhang et al. [26] proposed a CNN based ST models called ST-ResNet for predicting citywide crowd flows. There are two types of crowd flows: inflow and outflow, defined as the total traffic of crowds entering or leaving a region within a time period. They partitioned a city into identical square grids. Inflow and outflow at a time interval t are aggregated into each grid, then the citywide inflow (outflow) can be represented as a matrix. This representation resembles a frame in a video. The prediction problem is to predict future citywide crowd flows given past observations.

ST-ResNet is comprised of four sub-models: temporal closeness, temporal period, temporal trend, and external features such as weather and holidays. The CNN structure helps ST-ResNet to capture spatial dependencies. To capture temporal signals, the authors adopted a model fusion strategy: group the selected past observations into recent, near, and distant history. Three CNN sub-models were trained to represent information of recent past, short-term period, and long-term history, respectively. Similarly, to incorporate external features, the authors manually extracted features from weather and public events data, and fed them into a fully connected neural network. Finally, the outputs from the four submodels were fused together to make prediction (see Figure 3).

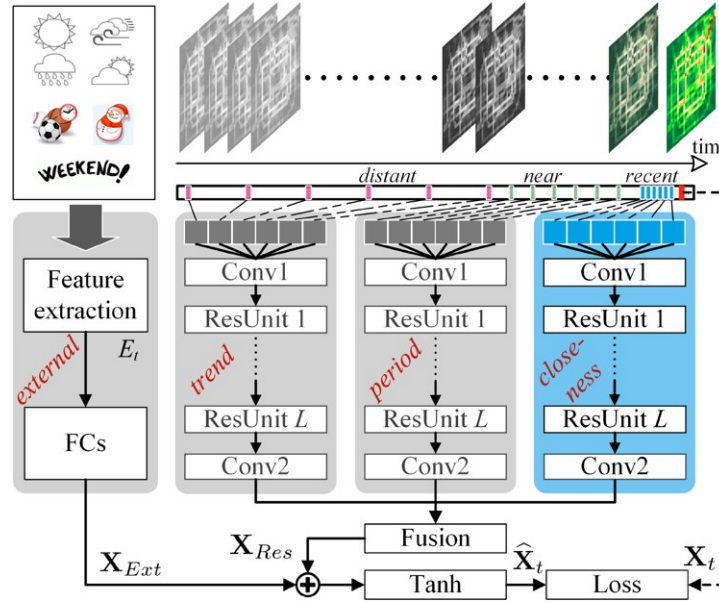


Figure 3: A CNN-based ST model consisting of four component: temporal closeness, temporal period, temporal trend, and external features [26].

The model is evaluated with Root Mean Square Error (RMSE) on two datasets: taxi GPS trajectories from Beijing and bike trajectories from NYC bike systems. Experiments show that ST-ResNet outperforms 9 baselines including ARIMA, LSTM, and DeepST [29]. Their results also show that LSTMs have particularly low accuracy when the input sequences is long (336), suggesting that LSTM cannot learn very long-term temporal dependencies. ST-ResNet overcomes this limitation of LSTM by using a model fusion technique at the price of higher model complexity and higher computation overhead. This example demonstrates the suitability of 2D CNN to model spatially continuous urban events.

Strengths of CNN based ST models

- These methods can capture local spatial context of neighboring regions without extra treatments.
- They are suited for prediction targets that are continuous over space. For example, predicting dockless bikeshare demand.

Limitations of CNN based ST models

- They need extra treatment to the model (e.g., model fusion) to capture temporal dependencies.
- They require image-like data representation, not suitable for network-like data. Determining the grid size is usually ad-hoc.

- They do not work well with extremely sparse data (e.g. predicting collisions at hourly interval for 1km by 1km regions). Networks with convolution structures may not work well due to the all zero weights caused by weight sharing of CNNs [30].

3.4 STNN models

CNN and RNN models rely on extra treatments to capture temporal or spatial context that their native architectures fail to. Researchers have attempted to marry both architectures in one framework. One way to achieve this goal is through model fusion [31, 32]. For example, Yao, et al. [3] proposed DMVST-Net for taxi demand prediction. The network consists of a CNN submodel, a LSTM submodel, and a graph embedding network to capture the latent semantics of regions. Although it is capable of reaping the benefits from both network structures, its performance heavily relies on careful model design. Instead of combining RNN and CNN at a submodel level, a more generic way to is to design a basic unit of a deep network that can learn spatio-temporal context. The rest of this section focuses on two examples of spatio-temporal neural networks: ConvLSTM [15] and 3D CNNs [7]. Both of them have been successfully applied to video prediction, speech recognition, and precipitation nowcasting [33].

3.4.1 ConvLSTM

ConvLSTM [15] adopts a LSTM network structure, but replaced fully connected nodes with convolutional structures in both input-to-state and state-to-state transitions, therefore achieving the advantages of both CNN and RNN. ConvLSTM takes three-dimensional input and outputs three-dimensional predictions (see Figure 4).

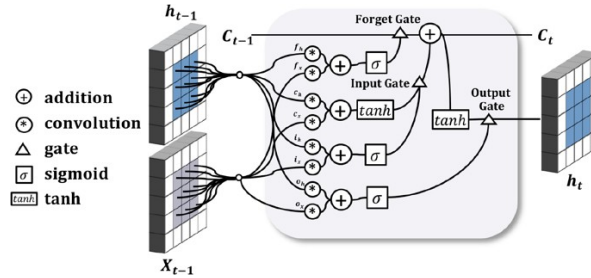


Figure 4: A ConvLSTM cell. Compared to LSTM cell, ConvLSTM replaced the inputs and outputs with convolutional structures [17].

Example: Traffic Accident Prediction

Yuan et al. [17] proposed a ConvLSTM variant called Hetero-ConvLSTM to predict the traffic accident of a region. Using Iowa State as a case study, they first partitioned the entire state into identical 5km by 5km square grids and aggregated the number of accidents into each grid for each day. The prediction task is to forecast the number of traffic accidents of the entire state for the next 7 days given the history of last 7 days.

The authors employed a feature-level fusion strategy to incorporate external information. Specifically, they collected features of various dimensions: 3D feature is the traffic volume of Iowa over the study period; 2D features include road network, road condition (e.g. number of intersections, number of lanes), and Google Earth satellite images; and 1D features include rainfall, temperature, and calendar features (e.g. holidays, month of year). As the road network is dense in urban areas and sparse in rural areas, traffic accidents exhibit outstanding spatial heterogeneity. To address this problem, the authors constructed a new set of 2D features that accounted for the spatial relationship between different areas based on road network. The new features were generated by constructing a spatial graph of the roads and performing eigen-analysis of the spatial graph. This procedure can be considered as learning an embedding for each road segment. All the features were concatenated with traffic accident data to form the input of Hetero-ConvLSTM (see Figure 5).

The model is evaluated with root mean squared error (RMSE), mean squared error (MSE), and cross entropy (CE). Experiments show that Hetero-ConvLSTM outperforms several classic machine learning methods, as well as deep learning based methods including LSTM and ConvLSTM. Results also show that the use of exogenous features in general improves prediction accuracy but some features may negatively affect accuracy.

This example demonstrates the effectiveness of ConvLSTM structure in modeling urban mobility events. However, ConvLSTM may inherit the limitations of LSTM such as inability to capture very long time dependencies. The

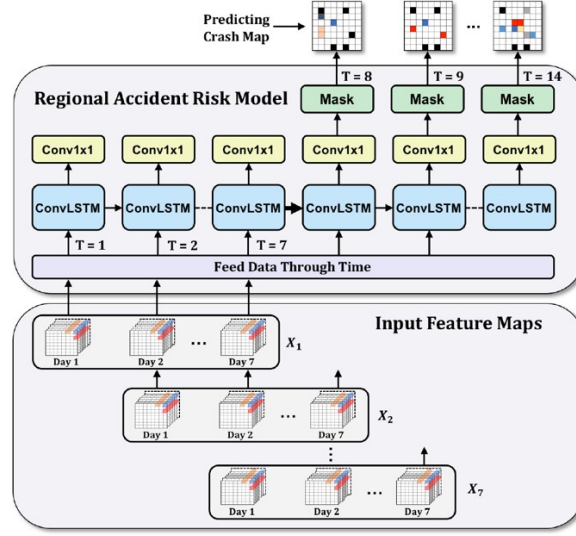


Figure 5: Predicting next 7 days traffic accidents based on the last 7 days history using Hetero-ConvLSTM [17].

length of input sequence of this example is 7 and the authors did not evaluate the cases that making predictions from longer history. Therefore, it is unclear whether Hetero-ConvLSTM will benefit or suffer from incorporating longer temporal information. Spatial heterogeneity is a challenge for State level predictions, but can be partially addressed by pooling operations in CNNs. Nevertheless, the LSTM’s chain-like structure limits ConvLSTM to perform pooling. Consequently, the spatial context learned by ConvLSTM is only local, constrained by the size of convolutional filters.

Advantages and limitations of ConvLSTM:

ConvLSTM inherited advantages of LSTM and 2D CNN. ConvLSTM can learn spatio-temporal dependencies simultaneously in one network. However, it can only learn local spatial context since spatial pooling cannot be applied to this structure. It cannot learn very long-term dependencies either, due to the limitation of LSTM. ConvLSTM requires 3D tensor-like input, therefore, it is not suitable for network data.

3.4.2 3D CNN

The core building block of 3D CNN is 3D convolution, which is capable of modeling spatio-temporal information. 2D CNN uses 2D convolution operations to compute features from spatial dimensions. As the input data is a 3D tensor like a video, 2D convolution will result in a 2D output without temporal information (see Figure 6). 3D convolution can preserve temporal information as the multiple contiguous frames of input are connected by the 3rd dimension of the filters, therefore the output is another 3D tensor containing temporal information [34].

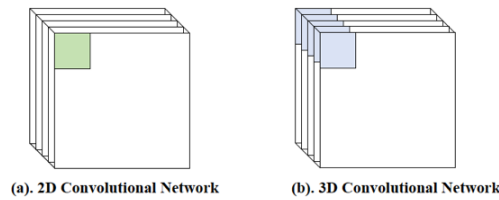


Figure 6: Comparison between 2D convolution and 3D convolution [16].

Example: Predicting taxi pick-up and drop-off

Shen et al. [16] proposed StepDeep, a network that uses 1D, 2D, and 3D convolutions to predict the number of taxi trips leaving and entering a certain region of a city at a certain time. Similar to Zhang et al.’s work [26] (see Section 3.3), the authors partitioned the city into square grids and aggregated the number of taxi trips leaving (pick-up) and

entering (drop-off) into each grid at each time step. In this way, the spatio-temporal taxi trips data can be represented by a 3D tensor. The prediction task is to forecast the next frame given selected historical frames that have strong temporal dependency (e.g. correspondence time in the past days, past weeks, and past months).

The main building blocks of StepDeep are 1D, 2D, and 3D convolutions (see Figure 7). 1D convolutions encode time series, 2D convolutions encode spatial information, and 3D convolutions encode spatio-temporal information. These three types of convolutional layers alternate and form 6 layers in total. The authors employed feature-level fusion to incorporate weather and holidays into training data.

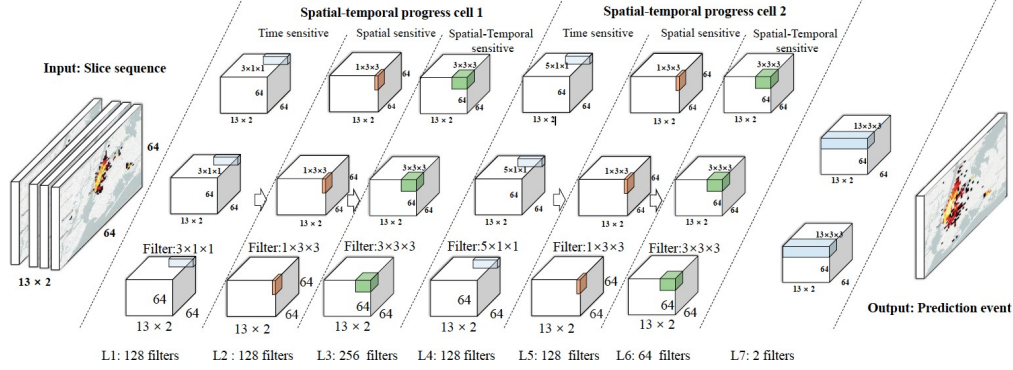


Figure 7: Predicting taxi pick-up and drop-off using 1D, 2D, and 3D convolutions. Exogenous information was stacked together with taxi data to form the input [16].

StepDeep was evaluated on NYC taxi dataset with RMSE. It outperformed five baselines including two conventional machine learning methods and a deep learning based method (i.e. DeepSD [35]).

Built on 3D convolutions, StepDeep is capable of capturing spatio-temporal patterns in one network. However, it does not use spatial pooling, so the spatial context learned is local, limited by the size of convolution filters. Another limitation is that this work only used temporal features such as weather. Additional 2D information such as road network and Point of Interest may help the model to achieve better performance.

Advantages and limitations of 3D CNN

3D CNN is able to learn local spatio-temporal patterns. Through pooling operations, it can capture longer spatial dependencies, which is an advantage over ConvLSTM. 3D CNN requires 3D tensor-like input, thus more suitable for spatially and temporally continuous data as compared to spatially discrete data or network data.

3.4.3 Strengths and limitation of STNN models

STNN models can capture spatio-temporal information in one network without additional features or modifications to model structure, so they are more generic than other types of deep learning architectures for spatio-temporal predictions. But compared to CNN or RNN-based models, STNN models are often more complex and have more parameters to learn. Consequently, they are data-hunger, expensive, and prone to over-fitting.

Another drawback of STNN is that the predictions tend to be “blurry” due to the “regression-to-mean” problem [36]. This is not specific to STNN but an limitation to spatio-temporal models that use l_2 loss (RMSE) or l_1 loss (MAE) as the optimization objective. Using the l_2 loss assumes that the data is drawn from a Gaussian distribution, which is often not true in mobility settings. Using l_1 loss suffers from the same problem but to a lesser degree. This problem is related to spatial heterogeneity problem: the predictions tend to ignore the mobility events that happen in remote areas or at lower frequencies. This is undesirable in demand prediction and traffic accidents prediction where underestimation may come at a higher price than overestimation. There are several ways to tackle this drawback: 1) Modifying the objective function by penalizing underestimations; 2) Taking advantage of General Adversarial Network (GAN) [36]; 3) Building separate models for different regions; and 4) Using features that encode spatial heterogeneity [17]. To date, this drawback of ST modeling has not been well-addressed by urban mobility research community.

3.5 Comparisons of different deep neural networks

The above sections described three types of deep learning architectures: RNN-based, CNN-based, and STNN models. Each of them has its own advantages and weaknesses and they can be combined with various data fusion techniques such as feature-level fusion and model fusion. In addition, unlike traditional AI fields such as computer vision and natural language processing, the adoption of deep learning is still new in urban mobility. As a result, well-acknowledged benchmark datasets and standard baselines are not available yet for researchers to evaluate different algorithms. More importantly, the tasks in urban mobility are often very different from each other in terms of the nature of the mobility event, data, and practical constraints. Taken together, it is presumptuous to conclude that one type of deep learning architecture or one combination of techniques is strictly superior than another for urban mobility applications. However, there are some general observations when comparing different deep learning architectures and data fusion techniques, which are summarized as follows:

Comparisons of different deep learning architectures

Task. Generally speaking, if the prediction target is continuous over space and time (e.g., predicting air quality, predicting dockless bikeshare demand) and can be represented as three-dimensional tensor like a video, STNN models have advantages over CNN and RNN, as STNN model architectures such as 3D CNN naturally capture spatial-temporal dependencies and thus do not require extra treatment such as constructing spatial-context features to model spatial or temporal dependencies. As a result, they usually perform better than CNN and RNN. If the task requires the model to capture signals from a long sequence of history, ConvLSTM may not be optimal as it cannot model very long sequences. If prediction target is discrete over space and has fixed locations of interest (e.g., predicting demand of each bikeshare station), RNN variants are sometimes more suitable than STNN models. If the prediction target is extremely sparse (e.g. predicting collisions at hourly interval for 1km by 1km regions), networks with convolution structures may not work well.

Data volume. The higher the model capacity, the more training data it needs. STNN models are more prone to over-fitting than simpler structures. The size of training data places an upper bound on the complexity of model structures we can use. When the size of training data is limited, consider reducing the number of parameters in a STNN model (e.g., reduce the number of hidden units in ConvLSTM) or switching to a simpler model structure.

Comparisons of different fusion techniques

Feature-level fusion is straightforward, and has lower computation cost than model fusion. Nevertheless, model fusion usually achieves better performance as it is based on the understanding of data sources and the model. These two techniques can be used together in one model.

The following example compares the performance of different approaches for new mobility resource demand prediction problems.

Example: new mobility resource demand prediction

I present two prediction tasks: predicting dockless bikeshare demand in Seattle and predicting ride-hailing demand in Austin (RideAustin²). Since these two tasks are similar, I focus on describing the Seattle bikeshare example.

The data used in this example contains more than 1,600,000 trips and more than 10,000 bikes in Seattle from October 1, 2017 to October, 31, 2018. I used the number of pickup (trip start) as a proxy for demand as there is no ground truth value for true demand. I partitioned the city into 20 by 32 squares, each representing a 1 km by 1 km region (Figure 8(a)). The number of bike pickup in each hour and in each square region based on pickup locations and timestamps is calculated. For each grid cell, bike demand forms a time series as shown in Figure 8(b). The prediction task is to forecast hourly demand for shared bikes for the City of Seattle based on the demand of last 7 days (168 hours). I generated slices of 169 hours as unit for training and prediction (168 hours for training and predicting next 1 hour). The training data contains 8040 temporal slices and the test data contains 1464 temporal slices.

I designed a prediction framework based on 3D CNN as it can automatically capture spatio-temporal context. I also included external features which are proven to be correlated with bikeshare demand according to existing literature [12, 37] to help with accuracy. Namely, 1D feature group includes temperature, precipitation, and sea level pressure; and 2D feature group includes bike lanes and steep slopes. I adopted a model fusion method to integrate these features into prediction. Specifically, I used three submodels consists of 1D, 2D, and 3D convolution layers to learn information from 1D features, 2D features, and 3D input (bikeshare), respectively. The output of each submodel was fused together, on top of which additional convolutional layers were applied to achieve the final prediction (See Figure 9). Compared to feature-level fusion as illustrated in Shen et al. and Yuan et al. 's work (See 3D CNN example and ConvLSTM example), this strategy has two main advantages: 1) Feature-level fusion treats different features equally, ignoring the

²<http://www.rideaustin.com/>

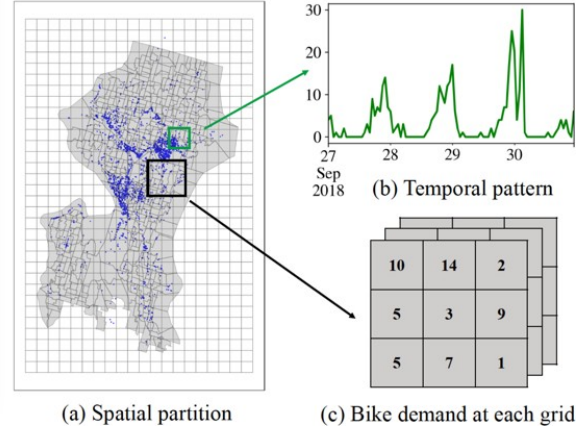


Figure 8: Data preprocessing. (a) We partition the city into square grids. (b) For each grid, bike demand forms a time series. (c) Each hour is akin to a frame in a video, with each grid cell as a pixel whose value is the demand.

semantic meaning of each feature. Nevertheless, features in 1D group represent mutually correlated meteorological information, and features in 2D group reflect the characteristics of the city. Therefore, feeding semantically related features into one submodel can potentially reinforce the effectiveness of one another. 2) Feature-level fusion requires 1D and 2D features to be duplicated into 3D. This procedure brings redundancy to the training data, which is further broadcast throughout the whole network, resulting in unnecessary computation overhead.

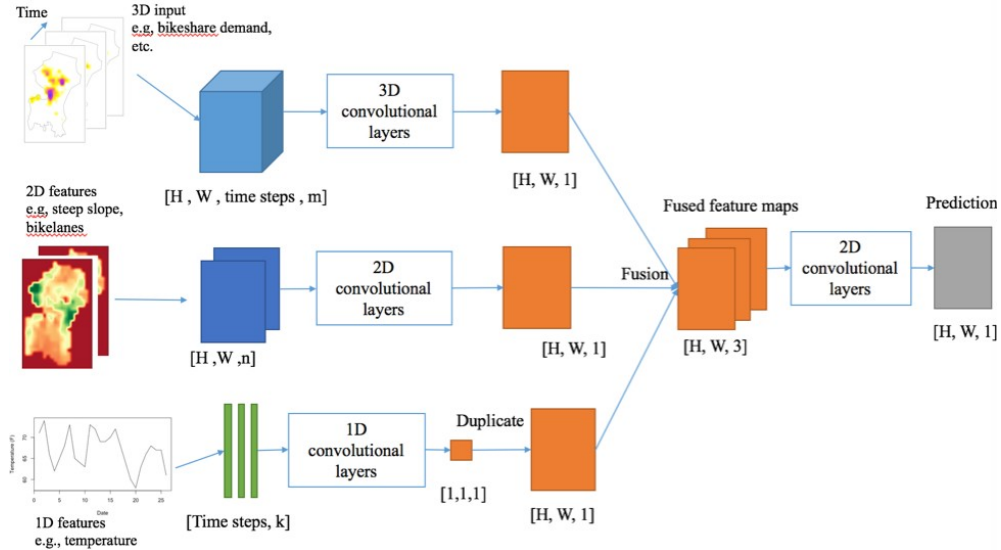


Figure 9: Use 1D, 2D, and 3D urban features to predict bikeshare demand in an end-to-end network. m , n , k are the number of 3D, 2D, and 1D features, respectively

The proposed model was evaluated using RMSE and MAE (mean absolute error) and compared with four baselines. 1) Historical Average (HA). I compute predictions of grid i using the mean values of all previous observations at location i at the same time of the day and the same day of the week; 2) Autoregressive Integrated Moving Average Model (ARIMA); 3) LSTM; and 4) ConvLSTM. I tested proposed model and baselines on both Seattle bikeshare data and RideAustin data. Table 1 shows the accuracy of all models on test dataset. It is observed that without any extra features, 3D CNN achieves higher prediction accuracy than the other four methods. LSTM achieves better accuracy than HA and AIRMA, but suffers from inability to learn information from spatial context. ConvLSTM performs the best among the baselines due to its capability of learning both spatial and temporal information. 3D CNN outperforms ConvLSTM since the 3D CNN architecture is more powerful in terms of capturing strong local spatio-temporal correlations in our

problem as compared to the recurrent architectures. Furthermore, the incorporation of 1D and 2D features improves accuracy in both Seattle bikeshare and RideAustin.

Table 1: Accuracy comparison

	Seattle Bikeshare		RideAustin	
	RMSE	MAE	RMSE	MAE
HA	2.1547	0.4838	4.3923	0.6612
ARIMA	3.5965	0.5383	3.3423	0.5973
LSTM	1.8314	0.4679	4.2555	0.5697
ConvLSTM	1.9107	0.4316	4.0250	0.5667
3D CNN	1.7436	0.4076	3.1415	0.5320
3D CNN + 1D	1.6879	0.3867	2.6995	0.4843
3D CNN + 1D + 2D	1.6654	0.3783	2.6597	0.4724

The “regression-to-mean” drawback of 3D CNN (ConvLSTM has the same drawback) is manifested with the visualization below. The prediction ignores the demand from south and northeast of the city.

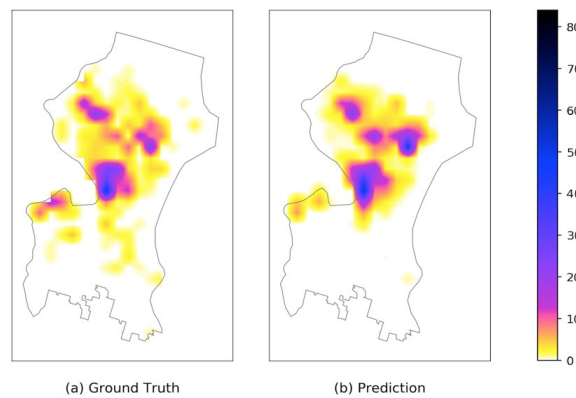


Figure 10: Ground truth and prediction of Seattle bikeshare demand at 2018-10-06 16:00:00. The prediction result ignores the demand from south and northeast of the city.

3.6 Other techniques

Besides the aforementioned model architectures and data fusion techniques, there are other techniques that can help address various problems in spatio-temporal modeling. For example, *Graph CNN* [25] can capture spatial-dependencies for network data [38]. *Attention Mechanism* [39] is used to help the model to focus only on the meaningful part of the spatial-temporal features [3]. These techniques can be used together.

3.7 Summary of Deep learning based ST modeling for Urban Mobility

Deep neural networks, combined with data fusion techniques are powerful in addressing two key challenges in urban mobility prediction tasks: capturing rich spatio-temporal correlations and extracting information from heterogeneous urban data. Recent urban mobility modeling research suggests that STNN models based on ConvLSTM or 3D CNN, generally yield better prediction accuracy than RNN or CNN based models for spatially and temporally continuous data [16, 17]. And it is generally agreed that the use of exogenous features from urban data sources helps prediction accuracy and model fusion helps achieve better prediction accuracy beyond feature-level fusion [18]. Nevertheless, the evaluation of various methods is dependent on the specific task and data. Sometimes trade-offs have to be made between accuracy and cost due to practical constraints. Furthermore, while various prediction frameworks have been proposed for urban mobility tasks, the generalizability of them outside their original application domains is unclear. Last but not least, standard baselines and large-scale benchmark datasets are needed to evaluate different methods from this rapidly evolving field.

4 Future Directions

Notwithstanding the promising examples of applying deep learning to address spatio-temporal predictions in urban settings, there are still many aspects that are missing or not well studied.

Improving spatial and temporal prediction accuracy. Architectures and techniques are needed to better capture long-term spatio-temporal dependencies, address spatial heterogeneity, and deal with sparse data.

Fairness-aware prediction for urban problems. Incorporating equity into prediction is crucial for many urban problems such as transportation resource distribution and crime prediction. However, fairness-aware methods for urban mobility is largely missing from current spatio-temporal machine learning literature. I made the first attempt to build a fairness-aware deep learning base model for bikeshare demand prediction by including a regularizer that enforces fairness into the objective function. Much more work is needed in this space.

Learning reusable and sharable features for urban mobility. The methods described in Section 3 are designed for individual urban mobility tasks. Nevertheless, we observe that these tasks share similar model structures and frequently use the same features such as transportation network and weather. Collecting and formatting these features take substantial efforts, and training separate models for each task is expensive in time and computation. It is therefore highly desirable if the knowledge (feature representation) learned from data through deep neural networks can be reused for other tasks or shared among multiple related tasks. A nascent thread of research focusing on spatio-temporal representation learning holds promises in deriving reusable and sharable knowledge from deep neural networks through multitask learning or unsupervised deep learning. For example, Li et al. [40] predicts origin-destination travel time along with several auxiliary tasks such as predicting travel distance and number of turns. Wang et al. [41] learned a spatio-temporal representation through deep autoencoder with an LSTM structure from GPS trajectories. This representation is used and reused for subsequent driving-related tasks such as predicting driving scores and identifying risky areas (See Appendix).

Learning transferable features for urban mobility. Machine learning assumes that the distribution of training data resembles that of test data. However, in real-world mobility settings, it is highly desirable to apply the knowledge learned from one city to new cities where training data is not available. Likewise, we may also want to apply the knowledge learned through one problem domain to related one in which training data is insufficient. For example, forecasting Uber demand from taxi demand. This problem has not yet well studied in urban mobility. Deep learning and the emergent Generative Adversarial Networks (GAN) provide new opportunities to achieve breakthroughs in urban mobility transfer learning. Among the very few research done in this domain, Liu et al. [42] presents an inspiring example of detecting shared bikes parking hot spots in a new city (see Appendix).

5 Limitations of Machine Learning in Urban Mobility

Machine learning methods are influential in urban mobility studies, nevertheless, they suffer from typical drawbacks of data-driven methods. First, models are learned to fit to the training data, so it is almost certain that the models will inherit and even amplify the biases in the data. The emergent field of fairness of machine learning aims to address this issue. Second, any knowledge gained through the modeling process is at best correlational not causal. Spurious correlations may exist and harm the generalizability of machine learning models. Furthermore, unlike domain-specific models (e.g. travel demand model), machine learning models are often designed without much consideration of domain knowledge or the guidance of domain-specific theories. One possible direction is to combine domain-specific models with machine learning, such as replacing sub modules of a process-based model which has weak theory basis with deep learning [13].

Compared to classical machine learning methods, there are several limitations of deep learning. Deep learning models are more sophisticated than classical machine learning models, and therefore much more training data is needed to learn the model parameters. Due to the same reason, deep learning models are prone to over-fitting and sensitive to hyper parameters. In practice, training deep learning models are still computationally expensive today. A laptop usually has sufficient computing power for most of the conventional machine learning algorithms but can rarely complete a deep learning task within reasonable time. More importantly, deep learning is less interpretable than classic machine learning approaches. For example, regression can explain the importance of hand-designed features through their coefficients. Yet deep learning extracts latent features automatically and it is often difficult to attach any semantic meaning to the learned features. Although feature importance can be qualitatively estimated through ablation studies, it is still far from establishing any causal relationship between features and prediction targets. Interpretability of deep learning is currently an on-going research field, a breakthrough of which may vastly increase the applicability of deep learning to a wider range of scenarios.

6 Conclusion

STNN models based on ConvLSTM or 3D CNN present a generic solution to modeling spatially and temporally continuous data. Combined with exogenous urban information, they are powerful in capturing spatial and temporal context for highly dynamic mobility systems, achieving significantly better accuracy than time series analysis and conventional machine learning methods. RNN-based and CNN-based models, along with data fusion techniques, can also effectively model the spatio-temporal relationship. Although they are less flexible in design, they have advantages over STNN models when considering the type of the prediction tasks and practical constraints such as computing cost and data availability.

I present several future research directions to address the challenges of current models including: 1) developing a better understanding of existing ST modeling methods and addressing spatial and temporal heterogeneity, as well as very long spatial and temporal dependencies; 2) learning reusable, shareable, and transferable representation for the use of multiple tasks; and 3) incorporating equity in ST modeling where relevant for social good, and exploring combining domain specific models with deep learning models for advancing science.

References

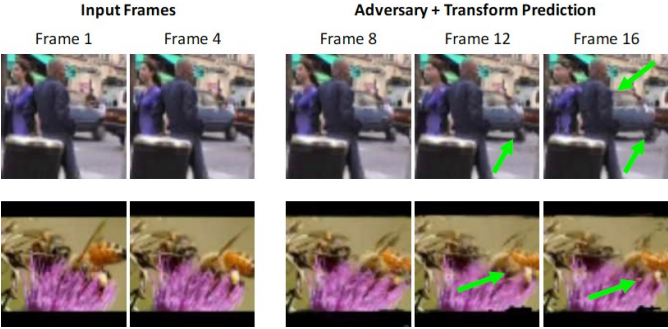
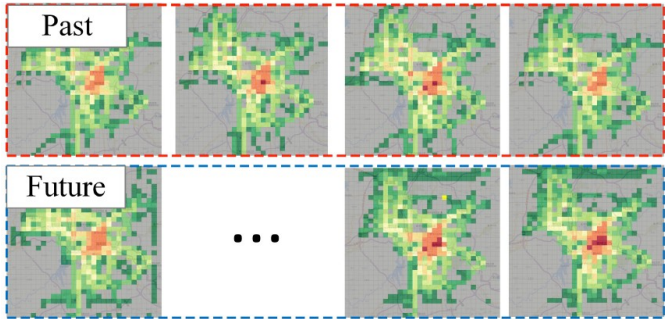
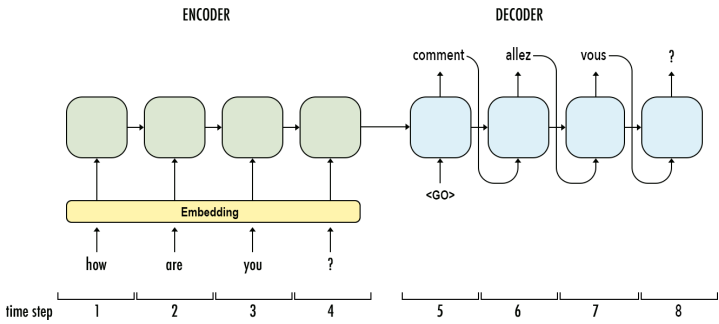
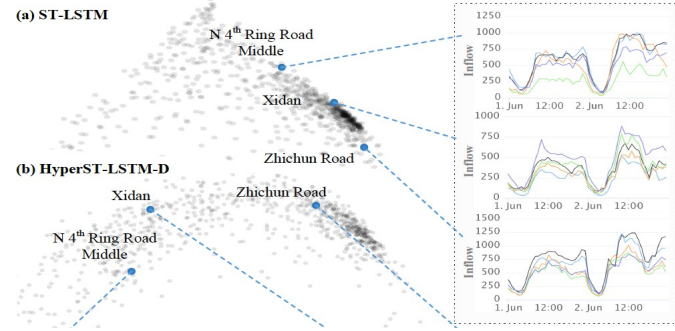
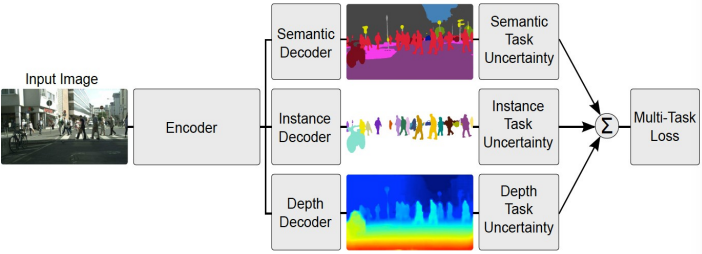
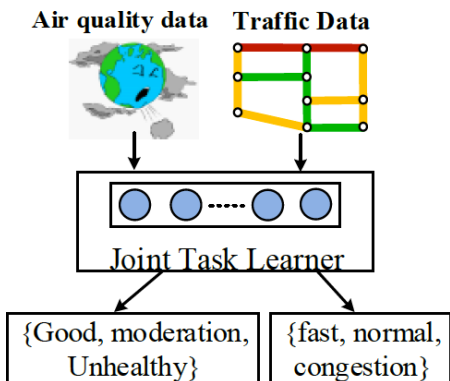
- [1] Chao Zhang, Keyang Zhang, Quan Yuan, Haoruo Peng, Yu Zheng, Tim Hanratty, Shaowen Wang, and Jiawei Han. Regions, periods, activities: Uncovering urban dynamics via cross-modal representation learning. In *Proceedings of the 26th International Conference on World Wide Web*, pages 361–370. International World Wide Web Conferences Steering Committee, 2017.
- [2] Ya-Fen Chan, Shou-En Lu, Bill Howe, Hendrik Tieben, Theresa Hoeft, and Jürgen Unützer. Screening and follow-up monitoring for substance use in primary care: an exploration of rural–urban variations. *Journal of general internal medicine*, 31(2):215–222, 2016.
- [3] Huaxiu Yao, Xianfeng Tang, Hua Wei, Guanjie Zheng, Yanwei Yu, and Zhenhui Li. Modeling spatial-temporal dynamics for traffic prediction. *arXiv preprint arXiv:1803.01254*, 2018.
- [4] Yu Zheng, Licia Capra, Ouri Wolfson, and Hai Yang. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):38, 2014.
- [5] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [6] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.
- [7] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.
- [8] Patrick Vogel, Torsten Greiser, and Dirk Christian Mattfeld. Understanding bike-sharing systems using data mining: Exploring activity patterns. *Procedia-Social and Behavioral Sciences*, 20:514–523, 2011.
- [9] Ji Won Yoon, Fabio Pinelli, and Francesco Calabrese. Cityride: a predictive bike sharing journey advisor. In *2012 IEEE 13th International Conference on Mobile Data Management*, pages 306–311. IEEE, 2012.
- [10] Luis Moreira-Matias, Joao Gama, Michel Ferreira, Joao Mendes-Moreira, and Luis Damas. Predicting taxi-passenger demand using streaming data. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1393–1402, 2013.
- [11] Pierre Hulot, Daniel Aloise, and Sanjay Dominik Jena. Towards station-level demand prediction for effective rebalancing in bike-sharing systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 378–386. ACM, 2018.
- [12] Yexin Li, Yu Zheng, Huichu Zhang, and Lei Chen. Traffic prediction in a bike-sharing system. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 33. ACM, 2015.
- [13] Markus Reichstein, Gustau Camps-Valls, Bjorn Stevens, Martin Jung, Joachim Denzler, Nuno Carvalhais, et al. Deep learning and process understanding for data-driven earth system science. *Nature*, 566(7743):195, 2019.
- [14] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.
- [15] SHI Xingjian, Zhouong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

- [16] Bilong Shen, Xiaodan Liang, Yufeng Ouyang, Miaofeng Liu, Weimin Zheng, and Kathleen M Carley. Stepdeep: A novel spatial-temporal mobility event prediction framework based on deep neural network. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 724–733. ACM, 2018.
- [17] Zhuoning Yuan, Xun Zhou, and Tianbao Yang. Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 984–992. ACM, 2018.
- [18] Yu Zheng. Methodologies for cross-domain data fusion: An overview. *IEEE transactions on big data*, 1(1):16–34, 2015.
- [19] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. Time-series extreme event forecasting with neural networks at uber. In *International Conference on Machine Learning*, number 34, pages 1–5, 2017.
- [20] Chengcheng Xu, Junyi Ji, and Pan Liu. The station-free sharing bike demand forecasting with a deep learning approach and large-scale datasets. *Transportation research part C: emerging technologies*, 95:47–60, 2018.
- [21] Rose Yu, Yaguang Li, Cyrus Shahabi, Ugur Demiryurek, and Yan Liu. Deep learning: A generic approach for extreme condition traffic forecasting. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 777–785. SIAM, 2017.
- [22] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
- [23] Bao Wang, Xiyang Luo, Fangbo Zhang, Baichuan Yuan, Andrea L Bertozzi, and P Jeffrey Brantingham. Graph-based deep modeling and real time forecasting of sparse spatio-temporal data. *arXiv preprint arXiv:1804.00684*, 2018.
- [24] Binbing Liao, Jingqing Zhang, Chao Wu, Douglas McIlwraith, Tong Chen, Shengwen Yang, Yike Guo, and Fei Wu. Deep sequence learning with auxiliary information for traffic prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 537–546. ACM, 2018.
- [25] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. In *International conference on machine learning*, pages 2014–2023, 2016.
- [26] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, Xiuwen Yi, and Tianrui Li. Predicting citywide crowd flows using deep spatio-temporal residual networks. *Artificial Intelligence*, 259:147–166, 2018.
- [27] Xiaolei Ma, Zhuang Dai, Zhengbing He, Jihui Ma, Yong Wang, and Yunpeng Wang. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, 17(4):818, 2017.
- [28] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.
- [29] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. Dnn-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, page 92. ACM, 2016.
- [30] Bao Wang, Penghang Yin, Andrea L Bertozzi, P Jeffrey Brantingham, Stanley J Osher, and Jack Xin. Deep learning for real-time crime forecasting and its ternarization. *arXiv preprint arXiv:1711.08833*, 2017.
- [31] Haiyang Yu, Zhihai Wu, Shuqin Wang, Yunpeng Wang, and Xiaolei Ma. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. *Sensors*, 17(7):1501, 2017.
- [32] Xingyi Cheng, Ruiqing Zhang, Jie Zhou, and Wei Xu. Deeptransport: Learning spatial-temporal dependency for traffic condition forecasting. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.
- [33] Feng Xiong, Xingjian Shi, and Dit-Yan Yeung. Spatiotemporal modeling for crowd counting in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5151–5159, 2017.
- [34] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.
- [35] Dong Wang, Wei Cao, Jian Li, and Jieping Ye. Deepdsd: supply-demand prediction for online car-hailing services using deep neural networks. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 243–254. IEEE, 2017.
- [36] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

- [37] Nathan McNeil, Jennifer Dill, John MacArthur, Joseph Broach, and Steven Howland. Breaking barriers to bike share: Insights from residents of traditionally underserved neighborhoods. 2017.
- [38] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017.
- [39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [40] Yaguang Li, Kun Fu, Zheng Wang, Cyrus Shahabi, Jieping Ye, and Yan Liu. Multi-task representation learning for travel time estimation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1695–1704. ACM, 2018.
- [41] Pengyang Wang, Yanjie Fu, Jiawei Zhang, Pengfei Wang, Yu Zheng, and Charu Aggarwal. You are how you drive: Peer and temporal-aware representation learning for driving behavior analysis. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2457–2466. ACM, 2018.
- [42] Zhaoyang Liu, Yanyan Shen, and Yanmin Zhu. Where will dockless shared bikes be stacked?:—parking hotspots detection in a new city. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 566–575. ACM, 2018.

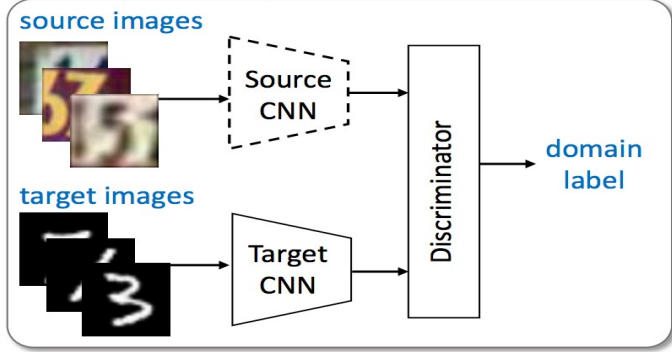
Appendix:

Table: Examples of deep learning applications and corresponding urban mobility tasks that use similar techniques.

Machine learning tasks	Urban mobility tasks
<p>Video prediction</p>  <p>Vondrick, Carl, and Antonio Torralba. "Generating the future with adversarial transformers." <i>Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition</i>. 2017.</p>	<p>Region-wide Forecasting</p>  <p>Zhang, Junbo, et al. "Predicting citywide crowd flows using deep spatio-temporal residual networks." <i>Artificial Intelligence</i> 259 (2018): 147-166.</p>
<p>Language translation</p>  <p>https://towardsdatascience.com/language-translation-with-rnns-d84d43b40571</p>	<p>Time series for flow prediction</p>  <p>Pan, Zheyi, et al. "HyperST-Net: Hypernetworks for Spatio-Temporal Forecasting." <i>arXiv preprint arXiv:1809.10889</i>(2018).</p>
<p>Multi-task learning for image</p>  <p>An architecture takes a single monocular RGB image as input and performs three tasks: classification, instance semantic segmentation and estimate of per pixel depth.</p> <p>Kendall, Alex, Yarin Gal, and Roberto Cipolla. "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics." <i>Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition</i>. 2018.</p>	<p>Shared representation: multi-task learning for cities</p>  <p>Co-predicts the air quality and traffic conditions at near future simultaneously.</p> <p>Zheng, Yu, et al. "Urban computing: concepts, methodologies, and applications." <i>ACM Transactions on Intelligent Systems and Technology (TIST)</i> 5.3 (2014): 38.</p>

Machine learning tasks

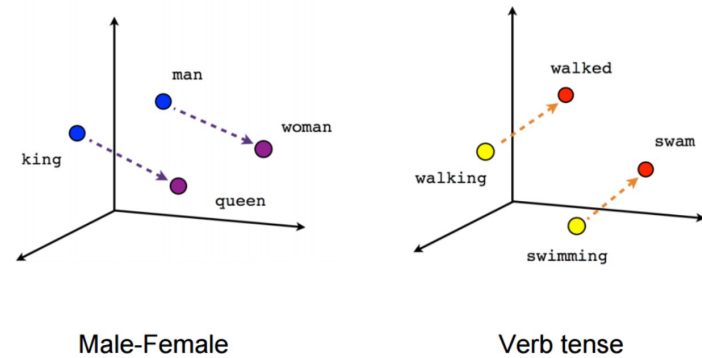
Domain adaptation (Transfer learning) for images



The knowledge learned from source images by the model to recognize digits can be transferred to classify target images.

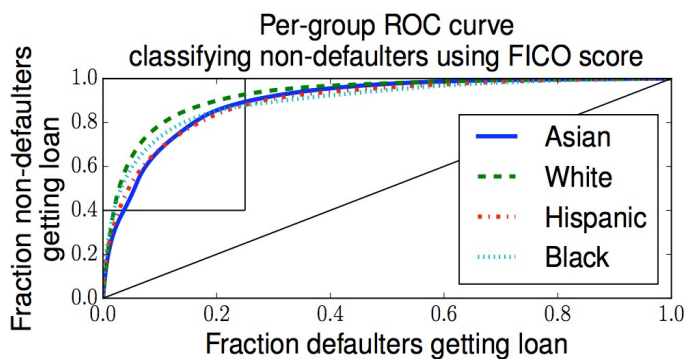
Tzeng, Eric, et al. "Adversarial discriminative domain adaptation." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

Word embeddings: reusable representation for language



<https://www.tensorflow.org/images/linear-relationships.png>

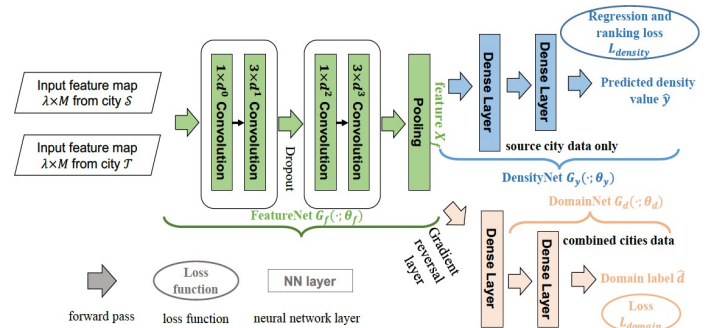
Fair prediction



Hardt, Moritz, Eric Price, and Nati Srebro. "Equality of opportunity in supervised learning." *Advances in neural information processing systems*. 2016.

Urban mobility tasks

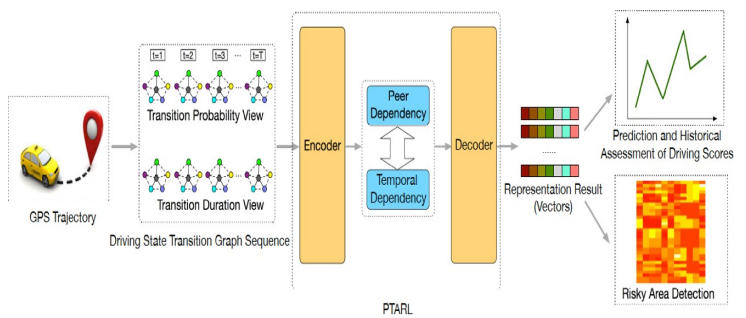
Knowledge transfer between cities or tasks



The knowledge learned through detecting dockless shared bike parking hotspots in one city can be used to detect hotspots in a new city where dockless bike systems have not been deployed.

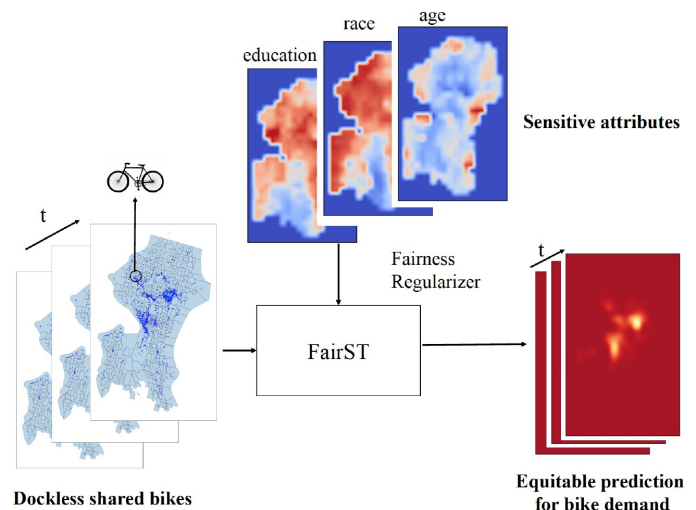
Liu, Zhaoyang, Yanyan Shen, and Yanmin Zhu. "Where Will Dockless Shared Bikes be Stacked?:---Parking Hotspots Detection in a New City." *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018.

Reusable ST presentation for urban mobility tasks



Wang, Pengyang, et al. "You are how you drive: Peer and temporal-aware representation learning for driving behavior analysis." *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018

Fair prediction for urban mobility



Enforcing fairness for dockless bikeshare demand prediction.