
Peyman Noursalehi, Haris N. Koutsopoulos, Jinhua Zhao

Abstract—Short-term demand predictions, typically defined as less than an hour into the future, are essential for implementing dynamic control strategies and providing useful customer information in transit applications. Knowing the expected demand enables transit operators to deploy real-time control strategies in advance of the demand surge, and minimize the impact of abnormalities on the service quality and passenger experience. One of the most useful applications of demand prediction models in transit is in predicting the congestion on station platforms and crowding on vehicles. These require information about the origin-destination (OD) demand, providing a detailed profile of how and when passengers enter and exit the service. However, existing work in the literature is limited and overwhelmingly focuses on forecasting passenger arrivals at stations. This information, while useful, is incomplete for many practical applications. We address this gap by developing a scalable methodology for real-time, short-term OD demand prediction in transit systems. Our proposed model consists of three modules: multi-resolution spatial feature extraction module for capturing the local spatial dependencies with a channel-wise attention block, auxiliary information encoding module (AIE) for encoding the exogenous information, and a module for capturing the temporal evolution of demand. The OD demand at time $t$, represented as a $N \times N$ matrix, is processed in two separate branches. In one branch we use the discrete wavelet transform (DWT) to decompose the demand into its different time and frequency variations, detecting patterns that are not visible in the raw data. In the other, three convolutional neural network (CNN) layers are utilized to learn the spatial dependencies from the OD demand directly. Instead of treating each channel of the resultant transformation equally, we use a squeeze-and-excitation layer to weight feature maps based on their contribution to the final prediction. A Convolutional Long Short-term Memory network (ConvLSTM) is then used to capture the temporal evolution of demand. The approach is demonstrated through a case study using 2 months of Automated Fare Collection (AFC) data from the Hong Kong Mass Transit Railway (MTR) system. The extensive evaluation of the model shows the superiority of our proposed model compared to the other compared methods.

Index Terms—Deep Learning, spatio-temporal modeling, wavelet decomposition, ConvLSTM, origin-destination demand

I. INTRODUCTION

In recent years, the proliferation of real-time data has led to a growing interest in utilizing the data to implement dynamic, adaptive strategies in advance of the demand surge to prevent unwanted deterioration in service quality. Such strategies include but are not limited to 1) adjust service headway or add service 2) disseminate information to passengers to expect a demand surge, which might incentivize some passengers to delay or change the time of their trip 3) implement crowd management strategies to avoid platform overcrowding [1]. Existing work in the literature overwhelmingly focus on short-term passenger arrivals at stations, often called “passenger flow”. However, while station arrival predictions are useful, it is the origin-destination (OD) demand information that is crucial for implementing many of the aforementioned strategies, as it provides a detailed profile of how and when passengers enter and exit the service. For example, real-time transit decision support system, which can be used for information dissemination and assessment of transit network scenarios [2], [3], rely heavily on accurate OD information. It has been shown [4] that platform and train crowding predictions that rely on the historical averages, as opposed to real-time predictions that utilize the latest information, can differ significantly from the ground truth, especially during the peak periods. This inaccuracy subsequently results in 1) inaccurate information communicated to passengers about the expected crowding levels on vehicles and hence damage the trust in the information, and 2) suboptimal control strategies. This underlines the importance of accurate OD demand predictions in real time.

In this paper, we propose a methodology for scalable, real-time, short-term OD demand prediction in transit systems. One of the key differences between passenger flow and the OD demand lies in the dimensionality of the problem. Assuming there are $N$ stations in a transit network, the OD matrix is of dimension $N \times N$ per time interval (e.g., 15 minutes). Furthermore, not only the OD demands from consecutive time intervals are correlated, they are influenced by the passenger arrivals at the origin stations. Most of the previous work on OD prediction has focused on developing univariate models, i.e., one model per OD flow. This approach however is not scalable and of limited practical use considering the high dimensionality of the problem. Inspired by the success of deep learning models in other fields, we propose an end-to-end deep neural network solution to this OD prediction problem. We summarize our contributions as follows:

P. Noursalehi is with the Department of Urban Studies and Planning, Massachusetts Institute of Technology, Cambridge, MA, 02139 USA (e-mail: peyman@mit.edu)

H.N.Koutsopoulos is with the Department of Civil and Environmental Engineering, Northeastern University, Boston, MA 02115, 02115

J. Zhao is with the Department of Urban Studies and Planning, Massachusetts Institute of Technology, Cambridge, MA, 02139 USA
1) We present a novel, scalable, deep learning architecture for real-time OD prediction in transit networks, using exit-based observations.

2) We propose a multi-resolution Spatio-Temporal neural network model (MRSTN) that captures the spatial and temporal dependencies, where a discrete wavelet transform is used to provide a multi-resolution decomposition of demand.

3) We test the proposed model on the entire Hong Kong Mass Transit Railway (MTR) system, and empirically show the superiority of our model over the other compared models.

The remainder of the paper is organized as follows. Section II provides a review of the previous work on demand prediction, including station arrivals and origin-destination flows. Section III describes the proposed methodology of MRSTN. In section IV, we present a case study where we apply our methodology using data from the MTR system and provide extensive experimental results. Finally, section V concludes our paper and discusses future work.

II. RELATED WORK

There have been many studies in the literature regarding short-term traffic flow prediction. [5], [6] provide an extensive review of the literature on this topic. The approaches can be broadly summarized into traditional time series models, in particular ARIMA models [7], [8], [9], linear regression models [10], [11], neural networks [12], [13], [14], [15], [16], nonparametric regression [10], [17], and state-space models [18], [19]. There is a growing literature on the short-term station arrival prediction methods. In [20], the authors use a Dynamic Factor Model for multivariate prediction of demand in the London Underground network. Recently, there has been an increased adoption of deep learning models, given their promising performance in other tasks, such as computer vision and natural language processing. In [21], the authors use Long short-term memory (LSTM) and fully connected layers (FC) to forecast the metro inbound/outbound passenger flow. Using a multi-scale radial basis function network, [22] proposes a methodology for predicting flow under atypical scenarios. In [23], the Beijing’s metro ridership (i.e. station arrivals) data is transformed into an image, which is subsequently processed by convolutional (CNN) and Bidirectional LSTM layers. Taking a different approach, the authors in [24] construct topological, similarity (based on Dynamic Time Warping distance), and correlation graphs to capture the dependencies among ridership of different transit stations. A variation of graph neural networks is then used for demand prediction.

We note, however, that all of the above models are developed for station arrivals, and not ODs. For an overview of deep learning models and their applications in transportation, the reader is referred to [25].

There have been numerous studies on short-term traffic origin-destination estimation and prediction [26], [27], [28], with a special focus on taxi and ridehailing trips ([29], [30]). There is, however, an important difference between traffic and transit OD prediction. In real-time, there is a time lag between the time a passenger enters at the origin station and exits at their destination. Hence, only partial data are available until the passenger completes their journey. In contrast, for traffic applications and station arrivals demand is completely observed. This is an important difference which we will address in section III, and provide a comparison with another approach suggested in the literature in section IV-G. For transit, OD predictions have been traditionally focused on planning applications. This may in part be because required data for OD prediction typically were not readily available. In some transit systems, passengers are not required to tap-out at their destinations. In such systems, OD data is not collected, and agencies have to use statistical techniques to infer destinations ([31], [32]). Another complication is the high dimensionality of the data; a network with N stations consists of N x N OD pairs, and the traditional methods of developing one model per OD pair is not scalable. [33] use Dynamic Bayesian Networks to obtain short-term train passenger flow forecasts between successive stations for a Paris metro line. They report that their models overall outperform forecasts based on historical averages. In a recent work, [34] proposed a hybrid model to accurately forecast the volume of passenger flows several steps ahead, but the model needs to be estimated and developed for each OD pair independently and separately. In contrast, our proposed model provides predictions for the whole network. The only other works, to the best of our knowledge, that explicitly address the delayed data collection issue are [35], [36]. Here, the authors use non-negative matrix factorization, based on incomplete entrance-based ODs to predict the true OD demand. In section IV-G we compare our approach to theirs and highlight the differences.

III. METHODOLOGY

A. Preliminaries

We represent the origin-destination matrix in each time interval \( \text{OD}_t \in R^{N \times N} \) as a two dimensional matrix, where \( N \) denotes the number of stations in the transit network. Specifically, \( \text{OD}_t(i,j) \) represents the demand from station \( i \) to station \( j \) that arrived at the origin station in time interval \( t \). It should be noted that the destination of a given trip is only observed upon its completion. In other words, the true \( \text{OD}_t(i,j) \) is only observed after a time lag that depends on travel time and transit schedule and can even be an hour or longer. This is one of the differences compared to the arrival prediction problem, where all arrivals up to the application of the prediction model are observed. We therefore, differentiate between the information available during the model estimation phase, and that in real-time. For model estimation, the true OD demand can be derived from a database of historical Automated Data Collection (AFC) transactions. Let \( \text{OD}_t(i,j) \) represent the demand originating from station \( i \) during time interval \( t \) heading to station \( j \). These trips might not have been completed till several time periods later. In real-time, however, we only have access to the OD as revealed by the exit counts. Let \( \text{OD}_{exit}^t(i,j) \) denote the exit-based OD matrix, where \( \text{OD}_{exit}^t(i,j) \) is the number of passengers that have exited station \( j \) during time interval \( t \) whose origin was station.
Our objective is to predict \( \text{OD}_{t+k}(i,j), k = 1 \ldots K \), from \( \text{OD}_{t-h}^{\text{exits}}, h = H \ldots 0 \), where \( k \) is the prediction horizon and \( H \) is the lookback window.

Our proposed model (MRSTN) consists of three modules, as shown in Figure 1: a multi-resolution spatial feature extraction module for capturing the local spatial dependencies, auxiliary information encoding module (AIE) to capture the impact of exogenous events, and a module for capturing the temporal evolution of demand. Each \( \text{OD}_{t-h}^{\text{exits}} \), represented as a matrix (image), is processed in two separate branches. In one branch, we use the discrete wavelet transform (DWT) to decompose the demand into its different time and frequency variations. This decomposition generates a richer representation of the input and has been a useful technique in many signal and image processing tasks ([37], [38]), and enables detecting patterns that are not visible in the raw data. Each sub-band is further processed by a convolutional neural network (CNN) to obtain its feature encoding. In the other branch, three convolutional neural network layers with skip-connections are utilized to learn the spatial dependencies from the OD demand directly. Additionally, a separate module (AIE) encode the auxiliary information, which includes passenger arrivals at the origin stations, weather conditions, and time of day. These feature maps are concatenated to provide a contextualized, multi-view representation of the original demand matrix. We then use a squeeze-and-excitation layer to weight feature maps based on their contribution to the final prediction. A Convolutional Long Short-term Memory network (ConvLSTM) is then used to capture the temporal evolution of demand. The resulting feature maps are subsequently passed through two convolutional neural networks. In the following sections we describe each component of our model in detail.

### B. Discrete Wavelet Transform

Discrete Wavelet Transforms (DWT) are a powerful tool for feature extraction in prediction tasks ([39], [40], [41]). This is in part because wavelets extend the concepts in Fourier transform to more general orthogonal bases, representing the input signal through a combination of small waves (hence the name, wavelet) that are localized in time and frequency. Wavelets have two important characteristics compared to Fourier transform: multi-resolution and orthogonality. Multi-resolution enables analyzing an image by zooming in and out, capturing different time and frequency variations and is useful for analyzing signals that arise from multi-scale processes [42]. Orthogonality means there is no redundancy between DWT channels [43]. The majority of previous studies have focused on univariate time series, and hence utilized the 1-dimensional DWT. A 1-D DWT decomposes any signal of one level into two sub-bands, approximation and detail, of another higher level. For a 2D image, the rows and columns are treated as 1D signals and the two passes at each round of the DWT are done at the rows and the columns separately. Typically, the coefficients of the sub-band are used to augment those from the original input, and then fed into a supervised learning algorithms.

Let \( I \) be the \( M \) by \( N \) input OD matrix. The 2D DWT decomposes the matrix into four sub-bands:

\[
\begin{align*}
\text{LL}(m,n) &= \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y) \varphi_{m,n}(x,y) \\
\text{LH}(m,n) &= \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y) \psi_{m,n}^H(x,y) \\
\text{HL}(m,n) &= \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y) \psi_{m,n}^V(x,y) \\
\text{HH}(m,n) &= \frac{1}{\sqrt{MN}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y) \psi_{m,n}^D(x,y)
\end{align*}
\]

where \( \varphi_{m,n}(x,y) \) is a scaling function of different scales and locations; \( \psi_{m,n}^H(x,y) \), \( \psi_{m,n}^V(x,y) \) and \( \psi_{m,n}^D(x,y) \) are the wavelet functions of different scales and positions. As a result, four sub-band images are obtained at each level, where three sub-band images LH, HL, HH are the detail images along horizontal, vertical and diagonal directions, respectively (Figure 2). The scaling function and wavelet functions are given by:

\[
\begin{align*}
\varphi_{m,n}(x,y) &= 2^{j/2} \varphi\left(2^j x - m, 2^j y - n\right) \\
\psi_{m,n}^H(x,y) &= 2^{j/2} \psi^H\left(2^j x - m, 2^j y - n\right), \quad i = \{H,D,V\}
\end{align*}
\]

where \( \varphi \) is the scaling function, \( \psi \) represents the wavelet functions; \( H, D \) and \( V \) represent the horizontal, vertical, and diagonal directions, respectively; and \( j \) is the decomposition level. DWT amplifies the features of different frequency sub-bands, enabling MRSTN to detect patterns that are not visible in the raw data. This, as we shown later in section IV, improves prediction performance significantly.

### C. Multi-resolution spatial feature extraction

The first layer utilizes convolutional neural networks to capture the spatial interactions. Each \( \text{OD}_{t-h}^{\text{exits}} \), \( h = H \ldots 0 \) is passed through two branches. In the first branch, it is passed through 3 CNN blocks, each comprising a convolutional layer, followed by a batch normalization layer and a ReLU activation function. The number of filters in the 3 CNN blocks are 64, 128, and 128, respectively, with kernel sizes of \( 3 \times 3 \). Since we want the output to have the same dimensionality as the input, we use stride of size 1 and no pooling. The output of this branch, \( F_1 \) is thus calculated as:

\[
F_1 = \text{Conv} (\text{OD}_{t-h}^{\text{exits}}, W_1)
\]

where \( W_1 \) denotes the weights of the convolutional layer. In a separate branch, a 1-level 2D-DWT decomposes \( \text{OD}_{t-h}^{\text{exits}} \) into four sub-bands to capture different time and frequency variations in the data (Figure 2): LL (approximation matrix), LH (horizontal matrix), HL (vertical matrix) and HH (diagonal matrix). In this paper, we use Daubechies 2 as the mother wavelet. Each sub-band is further processed by a CNN with 64, 3x3 filters with stride of 1, followed by a batch normalization...
Fig. 1: Overview of the proposed model. It consists of three modules: multi-resolution spatial feature extraction module for capturing the local spatial dependencies with a squeeze-and-excitation layer for channel-wise attention, auxiliary information encoding module (AIE), and a module for capturing the temporal evolution of demand (ConvLSTM).

layer and a ReLU activation function. The output of this branch, $F_2$, is thus:

$$F_2 = \text{Conv}(LL, W_2) \oplus \text{Conv}(HL, W_3) \oplus \text{Conv}(HL, W_4) \oplus \text{Conv}(HL, W_5)$$

where $\oplus$ denotes the concatenation operation. The learned feature maps from the two branches are then concatenated. These blocks together extract increasingly complex hierarchical demand patterns from local spatial dependencies among the OD pairs.

D. Auxiliary information encoding (AIE)

let $A_t \in \mathbb{R}^{N \times 1}$, $A_t = [a_{t,1}, a_{t,2}, \ldots, a_{N,t}]$ represent the $N$ dimensional column vector consisting of passenger arrivals at stations during time interval $t$, where $N$ denotes the number of stations in the transit network. We tile $A_t$ to be of the same shape as $OD_{t \times t}^{\times t}$, i.e., $N \times N$. The time of day $t$ and the weather condition are also tiled and concatenated along the spatial dimensions of the input tensor. We used the Dark Sky API [44] to extract the information on the weather conditions during the days of our case study. The conditions were categorized as normal, rainy, and snowy. We used the one-hot transformation to encode the conditions. These three inputs ($A_t$, time of day, and the weather conditions) were concatenated and passed through a convolutional neural network with 64 filters of size $3 \times 3$ and stride 1. Let $F^{Aux}$ denote the output of this transformation.

E. Squeeze and Excitation Layer

The outputs of all previous transformations are then concatenated together, providing a multi-view encoding of the information, i.e.:

$$F_3 = F_1 \oplus F_2 \oplus OD_{t \times t}^{\times t} \oplus F^{Aux}$$

$F_3 \in \mathbb{R}^{H \times W \times C}$ encodes all the available information in a tensor with $C$ channels. However, passing this tensor for any further convolutional operation will result in treating each channel with the same importance. Instead, we use the squeeze-and-excitation (SE) block proposed by [45] to learn the importance of each channel. This block effectively acts as a gated attention block, which learns to put more weight on channels that are more informative for the prediction task. Following the proposed mechanism in [45], we first squeeze the transformation output $F_3$ through its spatial dimensions, $(H, W)$, by using a global average pool, and generate channel-
wise statistics $z \in \mathbb{R}^C$, where the $c$-th element is calculated by:

$$z_c = F_{\text{squeeze}}(F_{3,c}) = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} F_{3,c}(i, j) \quad (6)$$

The main disadvantage of using convolutional filters for generating channel-wise statistics is that they only extract information within their local receptive fields. In contrast, the squeeze function exploits the contextual information outside the local receptive field. Having summarized each channel's information through the squeeze operation, an excite transformation $F_{\text{excite}}$ then acts on its output to capture the channel-wise interdependencies. Specifically, a simple gating mechanism with a Sigmoid activation function is applied as follows:

$$s = F_{\text{excite}}(z, W) = \sigma(g(z, W)) = \sigma(W_2 \delta (W_1 z)) \quad (7)$$

where $\sigma$ is the Sigmoid, and $\delta$ is the ReLU function. $W_1 \in \mathbb{R}^{2 \times C}$ parameterises a fully connected (FC) layer that reduces the input dimensions according to the reduction ratio $r$, and $W_2 \in \mathbb{R}^{C \times 2}$ denotes the parameters of a second FC layer that rescales the output to be of dimension $c$. These two FC layers effectively form a bottleneck layer that was shown to be effective in improving generalization and limiting model complexity. In this paper we use $r = 16$ as recommended in [45]. Finally, the output of the block is:

$$x_c = F_{\text{scale}}(F_{3,c}, s_c) = s_c \cdot F_{3,c} \quad (8)$$

where $F_{\text{scale}}$ denotes the channel-wise multiplication between the learned feature maps $F_{3,c} \in \mathbb{R}^{H \times W}$ and scalar $s_c$, and $X = [x_1, x_2, \ldots, x_C]$ represents the weighted output by the SE block.

F. Temporal feature extraction

A convolutional LSTM network is a variation of the LSTM model, introduced in [46]. The downside to using LSTMs for capturing the temporal dependencies is their inability of using the spatial information encoded in the input. ConvLSTMs address this problem by having convolutional structures in both the input-to-state and state-to-state transitions that can be trained end-to-end. Let $X_t \in \mathbb{R}^{H \times W \times C}$ denote the 3 dimensional input tensor at time $t$, computed according to equation 8. The ConvLSTM determines the future state of a certain cell in the grid by the inputs and past states of its local neighbors. At each time step $t$, the ConvLSTM unit receives three inputs: the past cell $C_{t-1}$, $X_t$, and the previous hidden state of the cell $H_{t-1}$. The input gate $i_t$ controls information accumulation from the its past inputs, the forget gate $f_t$ decides how much the past cell state $C_{t-1}$ should be forgotten, and finally, $O_t$ decides how much information from $C_t$ should be stored in the hidden state $H_t$. Mathematically, it can be written as:

$$i_t = \sigma(W_{xi} \cdot X_t + W_{hi} \cdot H_{t-1} + W_{ci} \odot C_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf} \cdot X_t + W_{hf} \cdot H_{t-1} + W_{cf} \odot C_{t-1} + b_f)$$
$$\tilde{C}_t = \tanh(W_{xc} \cdot X_t + W_{hc} \cdot H_{t-1} + b_c)$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$
$$o_t = \sigma(W_{xo} \cdot X_t + W_{ho} \cdot H_{t-1} + W_{co} \odot C_t + b_o)$$
$$H_t = o_t \odot \tanh(C_t) \quad (9)$$

where $\ast$ denotes the convolutional operator, $\odot$ is the hadamart-product, and $\sigma$ is the Sigmoid function.

The output of the ConvLSTM layer is subsequently transformed by two CNN blocks of 128 and 1 (i.e. the final prediction) filters, with stride one and no pooling layers.

G. Implementation details

We used Tensorflow [47] to implement our proposed method. We used the Adam optimizer [48] with initial learning rate of $10^{-4}$. The loss function is the mean absolute error between (MAE) the predicted demand and the ground truth. One reason for choosing MAE, instead of root mean squared error (RMSE), is that the latter penalizes large deviations more harshly than the former. Considering the distribution of demand profiles, where a small number of ODs attract the majority of demand, RMSE results in significantly worse predictions for less busy OD pairs. For DWT, we used the PyWavelet library in Python [49]. The model takes 2 hours to train using Google Collabatory with one T4 GPU.

IV. CASE STUDY

A. Data

We evaluated our proposed model using data from the Hong Kong Mass Transit Railway (MTR) system. The network consists of 97 stations. Transactions are recorded when passengers enter and exit the system, thus revealing the tap-in and tap-out stations and timestamps. We used two months of AFC transactions in October and November 2018, excluding the weekends, for a total of 40 days. The last 7 days were used for testing, and the rest for training the model. Each day was discretized into 15-minute bins, and the objective is to predict the OD demand for the next 15 minutes.

B. Preprocessing

To avoid gradient explosion and speed up the stochastic gradient optimization, the inputs should be normalized. During training, we normalize the OD demand for each origin-destination pair $i, j$ to be in the range of $[0, 1]$ using Min-Max normalization:

$$\text{OD}_{t,i,j}^{\text{exits}} = \frac{\text{OD}_{t,i,j}^{\text{exits}} - \min(\text{OD}_{t,i,j}^{\text{exits}})}{\max(\text{OD}_{t,i,j}^{\text{exits}}) - \min(\text{OD}_{t,i,j}^{\text{exits}})} \quad (10)$$
TABLE I: OD grouping based on their average demand per time interval

<table>
<thead>
<tr>
<th></th>
<th>High</th>
<th>Medium</th>
<th>Low</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg demand during the peak 15-min interval $OD_{i,j}$</td>
<td>$&gt; 250$</td>
<td>$250 \leq OD_{i,j} \leq 50$</td>
<td>$&lt; 50$</td>
</tr>
<tr>
<td>Number of OD pairs</td>
<td>42</td>
<td>198</td>
<td>9169</td>
</tr>
</tbody>
</table>

C. Evaluation Metrics

We use the Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) to evaluate the predictive performance of all models, defined as:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i^p - y_i^o|$$  \hspace{1cm} (11)

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (y_i^p - y_i^o)^2}{n}}$$  \hspace{1cm} (12)

where $y_i^p$ is the prediction, $y_i^o$ is the true, observed demand, and $n$ is the number of sample points.

D. Prediction accuracy

We categorize the OD demand into three categories (High, Medium, Low) based on their average demand during the peak 15-minute time interval to better assess the prediction accuracy (Table I). We compare the performance of our model with the following baselines:

- **Historical average (HA):** For each time interval $t$, we use $\frac{\sum_{m=1}^{m} OD_{i,j,t,m}}{n}$ as the predicted demand, where $OD_{i,j,t,m}$ is the demand between origin-destination pair $i,j$ on day $m$.
- **ARIMA:** We use the *auto.arima* function from [50] to automatically estimate an ARIMA model for each OD pair.
- **Support Vector Regression (SVR):** For each OD pair $i$ and $j$, we use $OD_{i,j,t-h}^{exit}$, $h = 5 \ldots 0$ to predict the true OD demand.
- **CNN:** Here we remove every component other than the upper branch, i.e., the 3 CNN layers from MRSTN.
- **LSTM:** We use a simple LSTM network with 64 units for each OD pair.
- **ConvLSTM:** We feed the input directly to a ConvLSTM layer.
- **ST-ResNet:** We use the architecture proposed by [51].
- **MRSTN - SE - DWT - Arrivals:** We remove the DWT decomposition, passenger arrivals at the origin stations, and the SE block from MRSTN.
- **MRSTN - SE - DWT:** We remove the DWT decomposition and the SE block from MRSTN.
- **MRSTN - SE:** The full MRSTN network without the SE block.
- **MRSTN - DWT:** The full MRSTN network without the DWT decomposition.
- **MRSTN:** The full MRSTN network.

The prediction performance on the test set is summarized in Table II for the next 15-min (1-step). The full MRSTN outperforms the others across all demand profiles. ST-ResNet is very competitive, especially for high demand ODs, although still outperformed by MRSTN. However, the joint effect of DWT and SE layers (the main differences between our architecture and ST-ResNet) is the substantially better predictions for medium and low demand OD pairs. This is not surprising; DWT decomposes the original input into multiple images, where some specifically capture the lower demand pairs (Figure 5). We further explore this observation in section IV-F. We note the importance of the SE block: its removal from MRSTN reduces the MAE of the results for the high demand stations by 3.2. Removal of the DWT component, on the other hand, reduces the MAE by 1.2. This observation suggests that when combining non-homogeneous information into one tensor, treating channels as equally important can lead to suboptimal results.

Figure 3 shows a few examples of the predicted versus the ground truth OD demand. An interesting observation can be made about the performance of the model for medium and low demand stations (top right and bottom, respectively). For these OD pairs, the demand between time intervals fluctuates significantly, resulting in a non-smooth, zig-zag pattern. MRSTN, however, learns to capture the underlying trend of the data, hence avoiding large penalties from following the zig-zag patterns. This is one of the reasons for its superior performance compared to the other models.

Figure 4 illustrates the evolution of the OD matrix across one day in the test set. Figure 4a illustrates the ground truth, and Figure 4b is the predicted OD matrix for the same time interval. Here, the busier an OD pair is, the closer its color is to red. We can see that the model can effectively capture the spatial and temporal demand patterns, especially for high and medium demand OD pairs.

E. Look-ahead horizon and the prediction accuracy

We further experiment to assess MRSTN’s performance on longer prediction horizons. Here, the objective is to predict $OD_{t+k}(i,j), k = 1 \ldots 4$, i.e., 15, 30, 45, and an hour into the future, from $OD_{t-h}^{exit}, h = 5 \ldots 0$. Table III provides a detailed breakdown of the long-term predictive performance. It can be seen that although the prediction accuracy drops as the prediction horizon expands, MRSTN still outperforms the historical average baseline in every demand category. For estimating the parameters of MRSTN for each of the long-term prediction tasks, instead of retraining the model from scratch, we initialize it using the learned weights for the 1-step ahead model and re-train. This results in more accurate predictive performance and faster training times, each taking fewer than 15 epochs to converge.

Table IV shows the running times of the various (deep learning) models. While MRSTN takes longer relative time to compute, the elapsed is still extremely small for all practical purposes.

F. Understanding the role of DWT

In this section, we provide some insights into the role of the DWT decomposition in the performance of MRSTN. As observed in Tables II and III, inclusion of DWT significantly
IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS

<table>
<thead>
<tr>
<th>Method</th>
<th>High demand</th>
<th>Medium demand</th>
<th>Low demand</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
</tr>
<tr>
<td>HA</td>
<td>25.8</td>
<td>42.1</td>
<td>5.9</td>
<td>19.8</td>
</tr>
<tr>
<td>ARIMA</td>
<td>24.1</td>
<td>41.4</td>
<td>6.4</td>
<td>21.1</td>
</tr>
<tr>
<td>SVR</td>
<td>28.1</td>
<td>46.2</td>
<td>6.9</td>
<td>21.5</td>
</tr>
<tr>
<td>CNN</td>
<td>27.1</td>
<td>46.5</td>
<td>6.5</td>
<td>21.0</td>
</tr>
<tr>
<td>LSTM</td>
<td>23.9</td>
<td>38.1</td>
<td>6.5</td>
<td>21.1</td>
</tr>
<tr>
<td>ConvLSTM</td>
<td>26.8</td>
<td>46.4</td>
<td>5.5</td>
<td>19.5</td>
</tr>
<tr>
<td>ST-ResNet</td>
<td>21.4</td>
<td>31.6</td>
<td>5.1</td>
<td>19.2</td>
</tr>
<tr>
<td>MRSTN - SE - DWT - Arrivals</td>
<td>23.4</td>
<td>32.2</td>
<td>5.0</td>
<td>19.3</td>
</tr>
<tr>
<td>MRSTN - SE - DWT</td>
<td>22.3</td>
<td>32.1</td>
<td>5.0</td>
<td>19.3</td>
</tr>
<tr>
<td>MRSTN - SE</td>
<td>21.1</td>
<td>31.3</td>
<td>4.9</td>
<td>19.2</td>
</tr>
<tr>
<td>MRSTN - DWT</td>
<td>19.1</td>
<td>28.5</td>
<td>4.8</td>
<td>19.2</td>
</tr>
<tr>
<td>MRSTN</td>
<td>17.9</td>
<td>27.6</td>
<td>4.7</td>
<td>19.1</td>
</tr>
</tbody>
</table>

TABLE II: Benchmarking the performance of MRSTN and the influence of its components

Fig. 3: Demand patterns for high (left), medium (right), and low demand OD pairs for one week in the test set. The red lines are the predicted result and the blue lines denote the ground truth.

Fig. 4: (a) Observed OD matrix at 7 am, 8 am, 9 am, 5 pm, and 7 pm for one day in the test set, from left to right. (b) 15-min ahead prediction of the OD matrix for the same time and day.
improves the prediction accuracy regardless of the prediction horizon. In Figure 5, we visualize the learned feature maps of the first CNN layer acting on the original (normalized) OD matrix (5b) and the 4 sub-bands of DWT decomposition (5c). It can be seen that in 5b, the learned feature maps try to mimic the input data, while in 5c there seems to be a separation of tasks. The feature maps learned from the approximation sub-band (top left of 5c) have a heavy focus on the busy OD pairs around the diagonal and the left hand side of the image. Compared to 5b, the features are more concentrated on the busiest stations. In contrast, the other 3 sub-bands seem to focus on the other OD pairs. Specifically, the features learned from the vertical and diagonal approximations mostly have zero weights for the busiest OD pairs and seem to be focused on the less busy stations. This is because of the orthogonality characteristic of the DWT decomposition, which means each sub-band is dedicated to learning a distinct feature of the input with minimal overlap. The results suggest that this multi-resolution approach enables the deep learning architecture to learn more fine-tuned features from the input image compared to the raw input.

G. Incomplete entrance-based OD versus exit-based
The authors in [35], [36] suggest using uncompleted entrance-based OD data in real-time. Specifically, at the current timestamp $t$, we can only observe trips that started at previous time intervals $t - h$ and the elapsed time has been sufficient for the vast majority of people to have arrived at their destination. Let $\mu_{i,j}$ and $\sigma_{i,j}$ be the mean and standard deviation of journey time distribution from station $i$ to $j$. Additionally, let $P_{OD,T}$ denote the indication matrix for all the complete entries of OD$_T$, where $P_{OD(i,j),T} = 1$ if the time horizon between $T$ and present time $t$ is longer than $\mu_{i,j} + 2\sigma_{i,j}$, i.e. we have approximately 98% confidence that all passengers have arrived at their destination if the journey times have a normal distribution. Then the true OD demand is calculated as $P_{\text{od},T} \otimes \text{OD}_T$. In Table V, we report the results from applying the input data as above and feeding it into MRSTN. Clearly the accuracy has deteriorated significantly. The journey time distributions between stations in our network (Figure 6) provide an explanation as to why this is the case. Assuming a lookback horizon of an hour, only about 50% of the trips have arrived at their destinations with 98% confidence (and therefore have $P_{OD(i,j),T} = 1$). Not only this results in an extremely sparse matrix, little real-time information is available to the model to adjust its predictions. In [35], [36] the authors adopt a sparse factorization method, which is suited for this representation of data, but it is not optimal for deep learning models.

V. Conclusion
In this paper, we propose a scalable methodology for real-time, short-term OD demand prediction in urban rail systems. Our model consists of three modules: multi-resolution spatial feature extraction module for capturing the local spatial dependencies, auxiliary information encoding module (AIE), and a module for capturing the temporal evolution of demand. We demonstrate the superior predictive performance of our model through a case study using 2 months of Automated Fare Collection (AFC) data from Hong Kong’s MTR system. We empirically show the importance of a multi-resolution analysis of the OD demand through DWT decomposition, as well as the channel-wise attention block. The proposed model can be used as a module of a predictive decision support system [4], enabling proactive control measures and advanced customer information generation.

ACKNOWLEDGMENT
The authors would like to thank Hong Kong Mass Transit Railway (MTR) for their support and data availability for this research.

REFERENCES
Fig. 5: (a) Normalized OD demand, (b) Feature maps learned directly from the original (normalized) OD matrix. (c) Learned feature maps from each one of the DWT decomposition sub-bands: approximation, horizontal, vertical and diagonal matrices (section III-C).

<table>
<thead>
<tr>
<th>Method</th>
<th>High demand</th>
<th>Medium demand</th>
<th>Low demand</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAE</td>
<td>RMSE</td>
<td>MAE</td>
<td>RMSE</td>
</tr>
<tr>
<td>Incomplete-OD MRSTN</td>
<td>24.2</td>
<td>35.3</td>
<td>5.3</td>
<td>19.5</td>
</tr>
<tr>
<td>Exit-based MRSTN</td>
<td>17.9</td>
<td>27.6</td>
<td>4.7</td>
<td>19.1</td>
</tr>
</tbody>
</table>

TABLE V: Benchmarking exit-based VS incomplete entrance-based ODs

Fig. 6: Cumulative distribution of journey times


**Peyman Noursalehi** is a Postdoctoral Associate at the MIT Transit and Urban Mobility Lab. He earned his PhD in Transportation Systems from Northeastern University, and his BSc in Civil Engineering from Sharif University of Technology.

**Haris N. Koutsopoulos** is a Professor in the Department of Civil and Environmental Engineering at Northeastern University in Boston, and research affiliate with the Transit Research Program at MIT. He also holds a position as Guest Professor of Transport Science at KTH, the Royal Institute of Technology in Stockholm, Sweden. He has extensive experience in modeling Intelligent Transportation Systems and developing simulation-based dynamic traffic assignment methods, traffic simulation models at various levels of resolution, simulation tools for the analysis of urban transit operations, and methods for processing and using data from opportunistic sensors for monitoring and control of transport system operations (traffic and transit). At KTH he established the iMobiity laboratory, The lab receives real time floating car and other traffic related data (e.g. traffic counts and speeds, weather), archived transit AFC and AVL data, and uses models to support applications such as monitoring and management of transport systems and multimodal travel planning services. For the work related to the iMobiity lab he received the IBM Smarter Planet Award in 2010.

**Jinhua Zhao** is the Edward H. and Joyce Linde Associate Professor of City and Transportation Planning at the Massachusetts Institute of Technology (MIT). Prof. Zhao brings behavioral science and transportation technology together to shape travel behavior, design mobility system and reform urban policies. He develops methods to sense, predict, nudge and regulate travel behavior, and designs multimodal mobility system that integrates autonomous vehicles, shared mobility and public transport. Prof. Zhao directs the Urban Mobility Lab (mobility.mit.edu) and the Mobility Initiative at MIT.