DST-Predict: Predicting individual mobility patterns from smartphone GPS data

SYED MOHAMMED ARSHAD ZAIDI¹, VARUN CHANDOLA², AND EUNHYE YOO³

¹Computer Science and Engineering, University at Buffalo-SUNY, Buffalo, NY 14260 USA (e-mail: szaidi2@buffalo.edu)
²Computer Science and Engineering, University at Buffalo-SUNY, Buffalo, NY 14260 USA (e-mail: chandola@buffalo.edu)
³Department of Geography, University at Buffalo-SUNY, Buffalo, NY 14260 USA (e-mail: eunhye@buffalo.edu)

Abstract: Predicting spatial behaviors of an individual (e.g., the number of times a person visits a specific location) is important to improve our understanding of the complexity of human mobility patterns, and to capture anomalous behaviors in an individual’s spatial movements, which can be especially useful in situations such as those induced by the COVID-19 pandemic. We propose a system called Deep Spatio-Temporal Predictor (DST-Predict), that can predict the future visit frequency of an individual based on one’s past mobility behaviour using the data collected from a cellphone. Predicting such spatial behavior is challenging, primarily because the patterns of location visits for each individual consists of both systematic and random components, which varies substantially across the spatial and temporal scales of analysis. To address these challenges, we propose a novel multi-view sequence-to-sequence model that uses Convolutional Long-short term memory (ConvLSTM) that makes use of past history of count visits features in order to predict the future count visits in a multi-step manner. Our experimental and results based on GPS survey data obtained from residents of a Western New York area in the United States show that the proposed system is capable of predicting the frequency of visits for individuals, in an urban setting, with high accuracy.

Index Terms: Human Mobility, Deep Learning, Predictive Learning

I. INTRODUCTION
Understanding human mobility patterns is important for solving problems related to public health [1], [2], emergency event detection [3], urban planning [4], and transportation engineering [5]. Literature [6] shows that daily human mobility patterns can be captured as a sequence of locations that individuals visited frequently, as they are the basis of essential travel activities such as going from home to workplace in the morning and coming back to home from work in the evening. Despite differences among individuals, previous studies [7], [8] have shown that large scale human mobility patterns are highly regular and predictable due to circadian patterns and routine daily activities, such as one’s journey to work or home.

Prediction of individual’s mobility over time, i.e., on an hourly, daily or weekly basis, enables us to better understand the general behavioral patterns of individuals and has been used in various practical applications, such as crowd flows prediction [9] and location-based advertising [10]. In the context of COVID-19 pandemic, a better understanding of current and future mobility patterns, especially in dense urban settings, is crucial for understanding and controlling the spread of the disease [11]. Predicting recurrent visits to a finite set of locations over time requires understanding of both spatial and temporal aspect of human movements. Previous studies [6] have demonstrated that mobility patterns can be captured by an exploration and preferential return model with a displacement distribution, in which individuals return to a limited number of places over time and their trips to places outside a regularly traveling region are rare. However, most previous studies are based on trajectory data extracted from mobile phone data logs, referred to as call detailed records (CDR), and focus only on large-scale mobility patterns. Prediction of individuals’ visit counts at frequently visited locations across multiple spatial and temporal resolutions have not yet been investigated.

GPS-enabled mobile phone data in which phone location is determined by special queries with pre-determined...
sampling intervals (“active mobile phone data” hereafter) have been increasingly used in human mobility studies [12]. A unique advantage of the active mobile phone data to other data modalities, such as CDRs or geo-tagged Twitter posts, which have been previously used to understand human mobility [13], is that they provide precise spatial location, compared to the closest antenna location for CDRs or the limited information present in geo-tagged tweets [14]. With a rapid increase in the availability of data arriving from heterogeneous sensors such as camera [15], loop detectors [16], [17], GPS [18], [19], etc., an opportunity has been offered for making use of deep learning approaches for developing novel and effective models that leverage the use of such huge amount of available data in order to provide useful solutions. Different models [20], [21] have been applied to different application use-case scenarios depending on the problem under study and the type of the corresponding data. Some of the related problems that applied the use of deep learning approaches include traffic flow prediction [22]–[24], traffic incident detection [25]–[27] and crime incident prediction [28], [29]. In the area human mobility pattern mining, deep learning approaches have been used in human behaviour analysis, estimating migratory flows and human trajectory data mining [30]–[32].

In this paper, we propose a system, DST-Predict that employs a novel multi-step deep learning architecture to: predict an individual’s visit frequency at a finite set of locations using the individual’s past active mobile phone data and other relevant information. The model architecture has been tested on different spatial resolution and has been shown to forecast visit counts many times ahead in the future. The model captures both spatial and temporal dependencies along with individual-specific characteristics, such as age, gender, employment status.

II. RELATED WORK

In this study, we investigate problem of visit frequency prediction, which can be considered as a special instance of count prediction problem explored in related contexts. The application of the predicted counts is wide, including from predicting crowd count in videos [33], predicting taxi demand [34], forecasting flow of crowds in a city [35] and tweet count prediction [36] within the specific geographic region. The other useful applications related to count prediction includes the counting in microscopic images [37], vehicle counting in images related to traffic congestion [38] and counting animals in the wild [39]. However, to the best of our knowledge, using GPS trace data to predict future visit frequencies has not been directly explored in the literature.

With the rapid increase in the use of deep learning approaches, the field of spatio-temporal data mining has witnessed a major change in solving different related tasks. Conventionally, deep learning algorithms have been used in the area of natural language processing [40], [41] and computer vision [42], [43], however, more recently these algorithms have been extensively used in the modeling and analysis of human mobility [33]–[35], [44]. Convolution neural networks (CNN) and Recurrent neural networks (RNN) have been used extensively in capturing spatial and temporal movements in human trajectory data mining studies [35], [45], [46]. The study [46] presented the use of both CNN and RNN to capture the spatio-temporal movements. Alternatively, a recent study [47] presented a unique convolution Long-short term network (ConvLSTM) for precipitation nowcasting on radar echo dataset to capture both the spatial and temporal correlation effectively. Specifically, some studies have used multiple machine learning techniques to count prediction problems in different settings, including using deep learning approaches to forecast crime incidents across different spatial-temporal scales. For example, [48] used deep learning in forecasting crime in a fine-grain city partition while [29] used ST-ResNet [35] to forecast crime distributions over the Los Angeles area. Deep learning approaches have also been used in understanding traffic flow and forecasting traffic accidents. For instance, Yuan et al., [49] used ConvLSTM on heterogeneous urban data for forecasting traffic accidents while Liu et al., [50] used ConvLSTM along with Bidirectional LSTM in predicting short-term traffic flow on the urban daily traffic data. Crowd counting is another problem in which several deep learning approaches have been employed in the past. For example, Zhang et al., [51] used deep convolution neural network to solve the cross-scene crowd problem while use of Bidirectional convLSTM for crowd counting in videos is presented in Xiong et al., [33]. However, the neural network architectures employed by these related solutions do not handle the unique challenges associated with the problem of visit frequency prediction from GPS data. These studies provided the forecasting at uni-timestep ahead which provided a limited outlook on the ability of the accuracy of the models. To overcome, some studies [24], [52] provided sequence-to-sequence based learning approach for problems related to traffic prediction. Our proposed solution handles these challenges using customized architectural and procedural modifications to perform the forecasting many timesteps ahead into the future.

III. PROBLEM FORMULATION

For each individual, the raw GPS data is available as a series of chronologically ordered GPS locations (latitude and longitude), denoted as $p_1^{(i)} \rightarrow p_2^{(i)} \rightarrow \ldots$, where the index in the superscript, $i$, denotes the $i^{th}$ individual. We transform this data into a gridded representation, by first grouping the locations by a user-specified temporal window, e.g., hourly, daily, weekly, etc. For each window, e.g., a day, we construct an $M \times N$ matrix $X_t^{(i)}$, where $M$ and $N$ are the number of rows and columns, respectively, of a uniform spatial grid of a particular resolution, applied on the target spatial area. $t$ denotes the index of the temporal window. Each entry of $X_t^{(i)}$ is equal to the number of times the $i^{th}$ user “visits” the corresponding grid cell, during the $t^{th}$ window. We will refer to the matrix $X_t^{(i)}$ as the visit count matrix for the $i^{th}$ user for
the $t^{th}$ time window.

Figure 1 illustrates this transformation for a sample user for the target area, as discussed in the subsequent sections. Note that, for this illustration, and for the remainder of this paper, we will use a daily window, which means that the predictions from DST-Predict will be obtained at a daily resolution, though the same methodology is applicable for any window length, depending on the target application.

In summary, the visit frequency prediction problem can be defined as follows: Given the historical visit count matrices until time $t$, denoted as $\{X^{(i)}_{f}\}_{f=t-1}^{t-d}$, predict the future visit count matrix $\{X^{(i)}_{f}\}_{f=t+1}^{t+d}$, where $f > t$ and $d$ is the forecasting time steps.

A. DST-Predict Overview

The DST-Predict system is a machine learning driven solution designed to solve the problem defined above. A process overview of the training and prediction steps for DST-Predict is provided in Figure 2. A training set, consisting of GPS trace data for a set of individuals is used in a bi-component approach to train the model to predict the future frequency count matrix, $X^{(i)}_{f}$, for a given spatial resolution (grid size) and window size. The prediction component of DST-Predict operates on an individual’s historical GPS data to make predictions at any time $t$ for a future time instance $f$.

The core engine of DST-Predict is a recurrent neural network based forecasting model, that can model the sequential/temporal dependencies in the data to make future predictions. A key aspect of the solution is that we treat the visit frequency matrix, $X^{(i)}_{t}$ as an image with $(M \times N)$ pixels. This allows us to utilize a convolutional architecture [42], which is the state-of-art approach to model the spatial relationships among the image pixels. However, the images in this context tend to be highly sparse, as shown in the distribution of the unique grid cells visited by each individual in the target urban area data set (See Figure 3). While there are 3200 possible grid cells, each user only visits approximately 15 grid cells on an average. On the other hand, a significant proportion of grids are visited by the individuals, with a strong spatial correlation, as shown in Figure 4. We handle this challenge by utilizing a loss function that can account for such sparsity in the data.

IV. METHODS

While the deep learning architecture of DST-Predict is described in Section IV-B, we first provide a brief overview of the individual components of the model in this section.

A. CONVOLUTION LONG SHORT-TERM MEMORY NETWORKS (CONV-LSTM)

As a widely used recurrent neural network, LSTM network can be used for solving sequence modeling problems. They are used for modeling temporal dependencies in sequence data, although existing algorithm uses a spatial aspect of the problem. To accommodate both the temporal and spatial dependencies in the data, Shi et al. [47] proposed the Convolutional LSTM (ConvLSTM) that is similar to fully connected LSTM (FC-LSTM) but uses convolution operator in the state-to-state and input-to-state transitions. The mathematical equations for the computations inside a ConvLSTM cell are as shown below:

\[
i_t = \sigma(W_{xi} \ast x_t + W_{hi} \ast h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \quad (1)
\]

\[
f_t = \sigma(W_{xf} \ast x_t + W_{hf} \ast h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \quad (2)
\]

\[
c_t = f_t \circ c_{t-1} + i_t \circ \tanh(W_{xc} \ast x_t + W_{hc} \ast h_{t-1} + b_c) \quad (3)
\]

\[
o_t = \sigma(W_{xo} \ast x_t + W_{ho} \ast h_{t-1} + W_{co} \circ c_t + b_o) \quad (4)
\]

\[
h_t = o_t \circ \tanh(c_t) \quad (5)
\]

where $\ast$ denotes the convolution operation and $\circ$ denotes the Hadamard (elementwise) product. Here, $i_t$, $f_t$, and $o_t$ are the outputs of the input, forget, and the output gate respectively. $c_t$ is the cell output at time step $t$ while $h_t$ is the hidden state of the cell at time step $t$. $\sigma(\cdot)$ is the logarithmic sigmoid function. $W_{xi}$, $W_{xf}$, $W_{hi}$, $W_{hf}$, $W_{xc}$, $W_{xo}$, $W_{ho}$, $W_{co}$ corresponds to the weight matrices. The usual meaning of each weight parameter matrix is indicated by the subscripts written alongside the symbol ($W$). For example, $W_{hi}$ is the weight matrix that maps the hidden to input gate. $b_i$, $b_f$, $b_c$, $b_o$ are the bias parameter matrices associated with input gate, forget gate, cell and output gate respectively. Also note that the input, $x_t$ is “flattened” into a $(M \times N)$-length vector, denoted by $x_i$. We have also dropped the superscript $(\cdot)^{(i)}$ from the notation, when referring to the data for a general user.

B. PROPOSED MODEL ARCHITECTURE

To account for the historical visit counts at different locations in different times, it is required to employ mechanism that handles both the spatial and temporal aspect of the data. We use ConvLSTM [47] as the basic unit in our architecture in order to handle these aspects in an effective manner. User visit count observations in the future does not only depends on the recent past history but also on far-distant history of observations. In order to effectively capture any complex spatio-temporal patterns within the visit counts of a user, we use two weeks of historical visit count observations for the first component, i.e. $p_1 = 14$ and one week history of historical visit count observations for the second component, i.e. $q_1 = 7$. In order to better capture both spatial and temporal dependencies, we propose the use of multi-component sequence-to-sequence architecture that consists of the following two components:

- **Component 1** uses past $p_1$ days of visit count data in the matrix format:

  \[
  X_{t-p_1}, X_{t-(p_1-1)}, X_{t-(p_1-2)}, \ldots, X_{t}
  \]
FIGURE 1: Transforming GPS trace data (left) to a gridded representation (right). Each grid cell is $2km \times 2km$ for this example.

FIGURE 2: Process overview of DST-Predict

- **Component 2** uses past $q_1$ days of visit count data in the matrix format:

  \[ X_{t-q_1}, X_{t-(q_1-1)}, X_{t-(q_1-2)}, \ldots, X_t \]

1) Component 1

For predicting the visit counts in a sequence, we adopt the approach of encoder-decoder architecture in **Component 1**, where the input sequence is processed and encoded into a latent vector of fixed length using one or many neural network layers. Here, this latent vector can be anticipated to be a good summary of the complete input sentence. The latent vector is then passed to the decoder phase where the decoder gets use this vector to start producing the output sequence using one or many neural network layers.

The input for this component first goes to the encoder ConvLSTM block shown in Fig. 6 that consists of three ConvLSTM layers in which the first two layers are followed by a Batch normalization (BN) layer, a non-linear LeakyRelu activation layer and a dropout layer. Batch normalization
FIGURE 3: Distribution of unique grid cells visited by each individual for the target data set. Each grid cell is $2km \times 2km$ and there are 3200 unique cells.

FIGURE 4: Number of visits for each grid cell ($1 km \times 1 km$) in the target spatial area (left). The urban counties are shown as shaded regions. The target geographical area is shown in the map on the right. 50% of the grid cells were visited at least once during the study period.

helps in reducing the internal covariance shift while speeding up the training process whereas LeakyReLU was employed to avoid the “dying ReLU” problems [53], [54] in training of deep neural networks. This dying ReLU problem arises when no gradient flow backwards so the neurons becomes inactive and only output 0 for any input. To avoid this issue, we use LeakyReLU activation layers instead of the other activation layers such as tanh, ReLU etc. Dropout [55] prevents overfitting issues as it provides the regularization in neural networks. The third ConvLSTM layer is just followed by a BN layer after which the output of the encoder is the encoded state vector that is passed to the decoder. In order to enhance the representational and learning power of the model in performing high level feature extraction from the inputs, we include a “shortcut” connection [56] that takes the output after the first ConvLSTM layer and adds it to the input for the final ConvLSTM layer. The purpose of these connections is to provide stability in training with stacking of more layers without leading to degradation of performance which may be caused due to vanishing/exploding gradients [57], [58]. The architecture components of the decoder is reverse similar to the encoder except that there is a final ConvLSTM layer at the end that gives the sequence of outputs of prediction values for future time steps. It is important to notice here that we transfer the last cell state $c$ (also called long-term memory) and the last hidden state $h$ (also called short-term
memory) from each of the ConvLSTM layers in encoder ConvLSTM block to all the ConvLSTM layers except the last ConvLSTM layer in the decoder block as shown in Fig. 6.

2) Component 2

For Component 2, we employed modifications in the overall architecture in comparison to the first component. Even though encoder-decoder architecture that we used in first component provides good results, it can still suffer from the problem of encoding a good summary of very long sequences as it is somewhat restricted into a fixed length of latent vector. In order to overcome this limitation, the use of attention mechanism [59] was proposed that enables to pay attention to each of the position of the input sequence while predicting output at each timestep. This makes use of the contribution or influence each data at each position in correspondence with each output.

The working principle of attention mechanism can be understood by first considering that if we have $T_x$ number of inputs in the sequence, then the annotations or hidden state outputs are denoted by $h_1, h_2, \ldots h_{T_x}$. In the simple encoder-decoder model, only the last state ($h_{T_x}$) of the encoder is used as the context vector and is then passed to the decoder, however, in attention mechanism [59], we compute the context vector $c_i$ for each target output $y_i$. Each of the context vector $c_i$ is generated using a weighted sum of annotations as given below:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \quad (1)$$

Here, the weight $\alpha_{ij}$ of each annotation $h_j$ is computed by

![Diagram of the proposed model architecture](image-url)
a softmax function given by the following equation:

\[
\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}
\]

(2)

where

\[ e_{ij} = a(s_{i-1}, h_j) \] (3)

is an alignment model that is responsible for scoring how well the inputs around position \( j \) and the output at position \( i \) match. It is important to note that the score here depends on the hidden state \( s_{i-1} \), which precedes the output \( y_i \) and the \( j \)-th annotation \( h_j \) of the input sequence.

In terms of architecture in encoder-decoder LSTM block for this component as shown in Fig. 7, the last ConvLSTM layer is followed by a Batch Normalization, Leaky ReLU activation and a dropout layer before feeding to the attention layer. Similarly, on the decoder side, right at the beginning we have a ConvLSTM layer followed by all these layers. One of the motivation for using these extra layers is to better capture high-level spatial features temporally which is best used by the attention layer that can better capture the past week’s input temporal sequence of the past week to generate the relevant output sequence for the next week.

In the overall model architecture, at last we have a final fusion layer that combines the sequence prediction coming from the two components to predict the final output sequence. We compute this output sequence by fusing the sequence outputs of different components of the model with associated learnable component weighted parameters as below:

\[
\hat{Y} = W_1 \odot \hat{Y}_1 + W_2 \odot \hat{Y}_2
\]

Here, \( \hat{Y}_1, \hat{Y}_2 \) are the predicted sequence output coming...
out of the two components of the model while $W_1, W_2$ are the trainable weight parameters that indicates the degree of influence that each of the component has on the final sequence prediction.

V. DATA AND EXPERIMENTAL SET-UP

A. DATA

The data used in the experiments was collected from a mobile phone application installed on the phones (Apple iPhone) of study participants. This application was used for collecting time-location information of movements of 1464 participants. The data has been carefully collected keeping under the consideration of the privacy of the users. The study area encompasses Buffalo-Niagara region within Erie and Niagara counties of western New York. Primarily, the data set comprises of the following information:

- **Demographics**: It comprises of the participants’ personal information such as gender, age, home and work address, employment status. In this study we only use the employment status as a user-specific feature. In this data set, approximately 17% of the individuals have a non-working status.
• **Global Positioning system (GPS) data**: It consists of the movement locations of participants collected at uniform temporal points through the application installed on their smartphones. The data was collected for a period of 32 weeks in the 2017-2018 time frame.

### B. EXPERIMENTAL SET-UP

In the data, there were missing data that needed to be handled before proceeding to the training phase. This included missing data for days in a sequence for a person. We imputed the values of the missing data with the mean value across each corresponding day of the week. The observed visit counts at each location is scaled to the range [0, 1]. While evaluating with the ground truth values, the prediction values are re-scaled back to the normal range. The experiments were conducted on a computing cluster available through Centre for Computational Research (CCR) in University at Buffalo. The nodes equipped with NVIDIA Tesla V100 GPUs with 16GB memory. We used Keras library [60] with Tensorflow library [61] as the backend.

1) **Model Training**

The total number of users for which we have data is 1464. Each user has GPS data records over a maximum period of 221 days (approx. 32 weeks), but some had less than 221 days. The average number of days for which GPS data has been recorded is about 179 days while the minimum of days that has been recorded in for any user is 53 days. Since only 17% of the 1464 users are having non-working status, we selected data for 485 out of 1464 users across 32 weeks in a way so that the total users indicated a well-balanced distribution of working and non-working status. We alternatively select the users based on this criteria i.e. out of 485 participants, every alternate participant has a non-working status. We train our model using the 80% of data for each of selected users. The model was validated with the remaining 20% data for each selected user.

2) **Choosing Hyperparameters**

In the Encoder ConvLSTM block for the first component, the first two Convolution LSTM (ConvLSTM) layers has 40 filters while the layer has 1 filter. The first ConvLSTM in the decoder block has 1 filter, the next two ConvLSTM layers has 40 filters while the final ConvLSTM layer has 1 filter. For the second component, all the ConvLSTM layers has 40 filters on both the encoder and decoder with the final ConvLSTM layer in the decoder having 1 filter. Each of the filter is of size $3 \times 3$ for extracting the relevant spatial features from both the input and output from the previous timesteps. Between each convLSTM layer we have employed batch normalization layer which is followed by Leaky ReLU and dropout layers. The dropout layer is set with the rate of 0.25. We train our model using the training data with batch size of 16 and 300 epochs. We used Adam [62] optimizer with learning rate of 0.001. The optimizer is set with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-07$ and clip value = 1.0. We also used model checkpoint that only saves the best weights while training.

### C. EVALUATION METRICS

To evaluate the predictive power of the proposed model to correctly identify the visit locations for a given individual, we need evaluation metrics that can measure the following two aspects:

1) **Recall** - What fraction of actual visits were correctly predicted by the model?

2) **Precision** - What fraction of the predicted visits corresponded to the actual visits made by the individual?

Mathematically, the two quantities can be calculated as follows. Consider a $(M \times N)$ test image matrix, $X$, at a given spatial resolution, and let $\hat{X}$ be the corresponding prediction matrix, obtained from the model. Note that we have dropped the time subscript, $t$, for clarity. The recall and precision are defined as:

\[
\text{recall} = \frac{\sum_{i,j=1}^{M,N} \min(X_{ij}, \hat{X}_{ij})}{\sum_{i,j=1}^{M,N} X_{ij}} \quad (1)
\]

\[
\text{precision} = \frac{\sum_{i,j=1}^{M,N} \min(X_{ij}, \hat{X}_{ij})}{\sum_{i,j=1}^{M,N} \hat{X}_{ij}} \quad (2)
\]

Note that, for both recall and precision, the numerator is the same and counts the overlap between the true and predicted visit counts for each grid cell. In this paper we report the average recall and precision over all daily visit counts matrices in the test data set.

An issue with the recall and precision metrics, as defined in (1) and (2), is that they are dependent on the spatial resolution (or scale) at which the matrices are created. Clearly, the task of predicting visit counts at a coarser resolution is easier than predicting visit counts at a finer resolution, and the expected recall and precision values at a coarser resolution will be higher than at finer resolution. Thus, the results obtained at different resolutions are not comparable.

This is a clear shortcoming in the present context, since we are interested in understanding the performance of the proposed model as a function of the spatial scale. To address this issue, we propose scale-invariant versions of the above defined recall and precision metrics.

We first calculate the recall and precision of a naive predictor, which simply distributes the total visits in $X$ uniformly across all the grid cells. The output of the naive predictor, denoted as $\tilde{X}$, is calculated as:

\[
\tilde{X}_{ij} = \frac{\sum_{i,j=1}^{M,N} X_{ij}}{M \times N} \quad (3)
\]

The base recall and precision for this naive predictor are defined as:

\[
\text{base\_recall} = \frac{\sum_{i,j=1}^{M,N} \min(X_{ij}, \tilde{X}_{ij})}{\sum_{i,j=1}^{M,N} X_{ij}} \quad (4)
\]

\[
\text{base\_precision} = \frac{\sum_{i,j=1}^{M,N} \min(X_{ij}, \tilde{X}_{ij})}{\sum_{i,j=1}^{M,N} \tilde{X}_{ij}} \quad (5)
\]
It is easy to verify that the values for the base_recall and base_precision metrics are expected to increase as the resolution becomes more coarse. This is because the probability of a randomly assigned visit to be placed in the correct grid cell by the naive predictor is \( \frac{1}{MN} \), which increases as the resolution becomes coarse, i.e., \( M \) and \( N \) become smaller. We use the performance of the naive predictor to “normalize” the recall and precision of the proposed model, as follows:

\[
\text{norm} \_ \text{recall} = \frac{\text{recall} - \text{base} \_ \text{recall}}{\text{base} \_ \text{recall}} \\
\text{norm} \_ \text{precision} = \frac{\text{precision} - \text{base} \_ \text{precision}}{\text{base} \_ \text{precision}}
\]

We report the normalized recall and precision values when comparing the performance of the proposed model across different scales.

1) Loss Function
Loss function, \( \mathcal{L} \) for training the model is composed of Mean Square Error (MSE) and square of the Mean Absolute Percentage Error (MAPE) and Structured dissimilarity (DSSIM). For a single training vector, \( \hat{Y} \) (a “flattened version” of the input \( I \times J \) matrix, where \( N = I \times J \)) and the corresponding prediction, \( Y \), the loss is defined as:

\[
\mathcal{L}(\theta) = \lambda_1 \frac{1}{N} \sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2 + \lambda_2 \frac{1}{N} \sum_{i=1}^{N} \left( \frac{Y_i - \hat{Y}_i}{Y_i} \right)^2 + \lambda_3 \text{MAPE} \]

Here, \( \theta \) are all the parameters that needs to be learned in the network. For the experiments, we choose \( \lambda_1 = 10 \), \( \lambda_2 = \lambda_3 = 1 \).

VI. RESULTS
We first discuss the results that summarizes the performance evaluation with impact of different spatial grid sizes and forecasting horizon during weekdays/weekends. We will then present the effectiveness of the model’s performance on the type of areas – rural and urban. Lastly, we present the comparative results for our model with state-of-the-art and competitive baseline approaches. We use the normalized recall and precision metrics, as defined in Section V-C, henceforth referred to as recall and precision.

A. IMPACT OF SPATIAL GRID SIZES AND FORECASTING HORIZON DURING WEEKDAYS AND WEEKENDS
In this section, we present the quantitative evaluation of results for the model on different spatial grid sizes – \( 2 \times 2 \), \( 3 \times 3 \), \( 4 \times 4 \) and \( 5 \times 5 \) and forecasting horizons. We tested our proposed model on 20% hold-out validation data for each of the 485 users. A comparison of the model performance at different forecast horizons is shown in Figure 9. A comparison of the model performance for one step ahead prediction between the weekdays and the weekends is shown in Figure 8. See Table 1 for a tabulated summary of results for different forecasting time horizon and spatial grid size.

As shown in Figure 9, the recall performance of the model is stable as the forecast horizon increases from 1 day to 7 days. The recall performance is best for a \( 4 \times 4 \) grid, and is worse for coarser as well as finer resolutions. However, for the precision evaluation metric, the model performance improves as the resolution becomes coarser, and is best for the \( 5 \times 5 \) grid. Moreover, for the finer resolution grids (\( 2 \times 2 \) and \( 4 \times 4 \)), the performance improves with the increase in forecasting horizon. Interestingly, precision remains stable while recall generally remains low for \( 3 \times 3 \) spatial grid size.

We can also notice an increase in performance as the forecasting time ahead increases. For \( 2 \times 2 \), there is a clear increase in performance in terms of all the evaluation measures with an increase in forecasting time ahead. For all other spatial grid sizes – \( 3 \times 3 \), \( 4 \times 4 \) and \( 5 \times 5 \), we can see that there is an increase in performance until third forecasting timestep, after which a slight decrease in performance can be clearly seen. This can be an expected result as generally the predictive power of a model decreases as the forecasting horizon is increased.

Figure 8 shows the performance of the model in forecasting during weekdays and weekends across different spatial grids. It can be clearly seen that across different spatial grid sizes, the model performs better in forecasting the visit counts during weekdays then compare to weekends. Moreover, the precision and recall increases as the grid size resolution goes from finer to coarser with an exception of recall for \( 3 \times 3 \). It can be seen that recall for this spatial grid size is more than any other spatial grid sizes.

B. PERFORMANCE EVALUATION FOR URBAN AND RURAL AREAS
Figure 10 shows the performance evaluation of the proposed model in urban and rural areas for a forecasting horizon of 7 days, with respect to different spatial grid sizes. The general trend for both areas is similar. However, the model is significantly better in predicting in urban areas than in rural areas. While part of this difference could be attributed to the fact that we have lesser data available for rural areas (see Figure 4), this is also indicative of the fact that the movement of individuals in urban areas is more predictive than in rural areas.

C. COMPARISON WITH OTHER COMPETITIVE AND BASELINE APPROACHES
Table 2 presents the performance evaluation of our proposed model with other approaches for \( 5 \times 5 \) spatial grid size at \( f = 7 \) using the data. We use the data for 10 users for training and testing. Here for each user, 80% is used as training data and the remaining 20% is used as validation data. The methods used for comparisons are:

1) ARIMA – Autoregressive integrated moving average (ARIMA), also known as Box-Jenkins model is a popular model used for time-series forecasting. It uses the historical time series for predicting future values in the series.
FIGURE 8: Evaluation of the model prediction for different grid sizes for $f = 7$ (left: Normalized Recall, right: Normalized Precision, higher values indicates better performance for both metrics). Results are shown separately for Weekdays and Weekends.

FIGURE 9: Evaluation of the model prediction for different forecast horizons (left: Normalized Recall, right: Normalized Precision, higher values indicates better performance for both metrics). Results are shown separately for different grid sizes.

FIGURE 10: Comparison of the model performance on urban and rural areas, for different grid sizes for $f = 7$ (left: Normalized Recall, right: Normalized Precision, higher values indicates better performance for both metrics).
2) **STResNet** [35] – State-of-the-art approach that makes use of convolutional layers and residual networks for spatio-temporal prediction.

3) **Res-ConvLSTM** [36] – STResNet based variant that makes use of ConvLSTM layers for spatio-temporal predictions.

4) **DST-Predict-Ext** – Here, we make use of incorporating external meta data such as weekday/weekend and employment status into our model. This was used to check whether there is any improvement in our prediction results if we fuse the external features into our model right after the fusion layer in a sequential manner.

5) **DST-Predict-without-Attention** – Here, we switch off the attention mechanism to check the performance of the model.

The results clearly indicates that our proposed model outperforms the state-of-the-art and competitive baseline approaches in terms of normalized recall and normalized precision.

**VII. CONCLUSIONS**

Our specific contributions are following. First, we propose a system, **DST-Predict** which uses a sequence to sequence deep learning approach to predict the visit frequency of an individual, based on the historical GPS location data. Second, we propose a customized loss function that takes variable weighted Mean Squared error (MSE), Mean absolute percentage error (MAPE) and Structured Similarity (SSIM). Third, we propose a scale-invariant evaluation metrics that effectively compares the performance of the model with respect to different spatial grid sizes. Lastly, experimental results on real GPS traces obtained for over 485 individuals over a period of 32 weeks for a Western New York area in United States indicates that the proposed system effectively forecast the visit counts for future forecasting horizons.

One motivation for this system was to test how accurately can we predict an individual’s future mobility behavior, based on past data, and our experimental results show that the model can indeed produce highly accurate predictions at different spatial resolutions. This task is challenging and the proposed deep learning architecture, handles the various modeling challenges, through specific customization, including the use of a residual block and a specialized loss function. Given the need for accurate mobility predictions for a variety of important applications, including understanding impact of mobility on spread of infectious diseases, understanding the privacy implications of mobile tracking, etc., the **DST-Predict** system can provide a vital predictive capability. One of the shortcomings of **DST-Predict** is the lack of geographic awareness in the model training. Each visit frequency matrix is treated as an image, and loses any geographic information, such as presence of water bodies and other hazards, when making the predictions. In future, we plan to develop customized loss functions that can explicitly incorporate such constraints into the model.

**ACKNOWLEDGMENT**

This work was partially supported by the National Science Foundation, Office of Advanced Cyberinfrastructure - OAC 1910539. Computing facilities were provided by the Univer-
TABLE 2: Performance evaluation of our proposed model (in bold) in comparison with other approaches. Each value represents mean ± stdev. This comparison is done using the GPS data for 10 users for 5 × 5 grid size at f = 7.

| Model                          | Norm. Precision | Norm. Recall |
|-------------------------------|-----------------|--------------|
| ARIMA                         | 0.14 ± 0.16     | 0.06 ± 0.15  |
| STResNet [35]                 | 0.25 ± 0.23     | 0.10 ± 0.14  |
| Res-ConvLSTM [36]             | 0.23 ± 0.24     | 0.14 ± 0.20  |
| DST-Predict-Ext               | 0.22 ± 0.22     | 0.17 ± 0.22  |
| DST-Predict-without-Attention | 0.50 ± 0.15     | 0.29 ± 0.14  |
| DST-Predict                   | 0.58 ± 0.13     | 0.31 ± 0.15  |

REFERENCES

[1] A. Alberdi, A. Weakley, M. Schmitter-Edgecombe, D. J. Cook, A. Aziria, A. Basarab, and M. Barrenechea, “Smart home-based prediction of multidomain symptoms related to alzheimer’s disease,” IEEE journal of biomedical and health informatics, vol. 22, no. 6, pp. 1720–1731, 2018.
[2] J. M. Barros, J. Duggan, and D. Rebolb-Schulmann, “Disease mentions in airport and hospital geolocations expose dominance of news events for disease concerns,” Journal of biomedical semantics, vol. 9, no. 1, p. 18, 2018.
[3] K. Gray, D. Smolyak, S. Badirli, and G. Mohler, “Coupled igmm-gans for deep multimodal anomaly detection in human mobility data,” arXiv preprint arXiv:1809.02778, 2018.
[4] F. Xia, J. Wang, X. Kong, Z. Wang, J. Li, and C. Liu, “Exploring human mobility patterns in urban scenarios: A trajectory data perspective,” IEEE Communications Magazine, vol. 56, no. 3, pp. 142–149, 2018.
[5] Z. Huang, X. Ling, P. Wang, F. Zhang, Y. Mao, T. Lin, and F.-Y. Wang, “Modeling real-time human mobility based on mobile phone and transportation data fusion,” Transportation research part C: emerging technologies, vol. 96, pp. 251–269, 2018.
[6] C. Song, T. Koren, F. Wang, and A.-L. Barabási, “Modelling the scaling properties of human mobility,” Nature Physics, vol. 6, no. 10, p. 818, 2010.
[7] A. Cuttone, S. Lehmann, and M. C. González, “Understanding predictability and exploration in human mobility,” EPJ Data Science, vol. 7, no. 1, p. 2, 2018.
[8] H. Barbosa, M. Barthelemy, G. Ghoshal, C. R. James, M. Lenormand, T. Louail, R. Menezes, J. J. Ramasco, F. Simini, and M. Tomasin, “Human mobility: Models and applications,” Physics Reports, vol. 734, pp. 1–74, 2018.
[9] R. Jiang, X. Song, D. Huang, X. Song, T. Xia, Z. Cai, Z. Wang, K.-S. Kim, and R. Shibasaki, “Deepurbanevent: A system for predicting citywide crowd dynamics at big events,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 2019, pp. 2114–2122.
[10] P. Palos-Sanchez, J. R. Saura, A. Reyes-Menendez, and J. V. Esquivel, “Users acceptance of location-based marketing apps in tourism sector: An exploratory analysis,” Journal of Spatial and Organizational Dynamics, vol. 6, no. 3, pp. 258–270, 2018.
[11] M. U. G. Kraemer, C.-H. Yang, B. Gutierrez, C.-H. Wu, B. Klein, D. M. Zaidi, S. Hadavi, H. B. Rai, S. Verlind, H. Huang, C. Macharis, and T. Guns, “Analyzing passenger and freight vehicle movements from automatic number plate recognition camera data,” European Transport Research Review, vol. 12, pp. 1–17, 2020.
[12] A. Ariannezhad and Y.-J. Wu, “Large-scale loop detector troubleshooting using clustering and association rule mining,” Journal of Transportation Engineering, Part A: Systems, vol. 146, no. 7, p. 04020064, 2020.
[13] S. M. A. Rizvi, A. Ahmed, and Y. Shen, “Real-time incident detection and capacity estimation using loop detector data,” Journal of Advanced Transportation, vol. 2020, 2020.
[14] Q. Yu, H. Zhang, W. Li, X. Song, D. Yang, and R. Shibasaki, “Mobile phone gps data in urban customized bus: Dynamic line design and emission reduction potentials analysis,” Journal of Cleaner Production, vol. 272, p. 122471, 2020.
[15] H. Allahbakhshi, L. Conrow, B. Naimi, and R. Weibel, “Using accelerometer and gps data for real-life physical activity type detection,” Sensors, vol. 20, no. 3, p. 588, 2020.
[16] G. Atluri, A. Karpatne, and V. Kumar, “Spatio-temporal data mining: A survey of problems and methods,” ACM Computing Surveys (CSUR), vol. 51, no. 4, pp. 1–41, 2018.
[17] S. Wang, J. Cao, and P. Yu, “Deep learning for spatio-temporal data mining: A survey,” IEEE Transactions on Knowledge and Data Engineering, 2020.
[18] S. Du, T. Li, X. Gong, and S.-J. Hong, “A hybrid method for traffic flow forecasting using multimodal deep learning,” arXiv preprint arXiv:1803.02099, 2018.
[19] W. Huang, G. Song, H. Hong, and K. Xie, “Deep architecture for traffic flow prediction: deep belief networks with multitask learning,” IEEE Transactions on Intelligent Transportation Systems, vol. 15, no. 5, pp. 2191–2201, 2014.
[20] B. Liao, J. Zhang, C. Wu, D. McIlwraith, T. Chen, S. Yang, Y. Guo, and F. Wu, “Deep sequence learning with auxiliary information for traffic prediction,” in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 537–546.
[21] H. Ren, Y. Song, J. Wang, Y. Hu, and J. Lei, “A deep learning approach to the citywide traffic accident risk prediction,” in 2018 21st International Conference on Intelligent Transportation Systems (ITSC), IEEE, 2018, pp. 3346–3351.
[22] Z. Zhang, Q. He, J. Gao, and M. Ni, “A deep learning approach for detecting traffic accidents from social media data,” Transportation research part C: emerging technologies, vol. 86, pp. 580–596, 2018.
[23] L. Zhu, F. Guo, R. Krishnan, and J. W. Polak, “A deep learning approach for traffic incident detection in urban networks,” in 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018, pp. 1011–1016.
[24] L. Duan, J. Zhang, C. Xu, and S. V. Scarpino, “The effect of human mobility and control measures on the citywide traffic accident risk prediction,” in 2020 IEEE 6th International Conference on Smart City, Technology and Sustainability (ICSC), 2020, pp. 1–5.
[25] R. Jiang, X. Song, Z. Zhu, J. Chen, Q. Chen, and R. Shibasaki, “Deep roi-based modeling for urban human mobility prediction,” Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 4, no. 3, p. e12252, 2020.
[26] L. Duan, T. Hu, E. Cheng, J. Zha, and C. Gao, “Deep convolutional neural networks for spatiotemporal crime prediction,” in Proceedings of the International Conference on Knowledge and Engineering (IKE). The Steering Committee of The World Congress in Computer Science, Computer, 2017, pp. 61–67.
[27] B. Wang, D. Zhang, D. Zhang, P. J. Brantingham, and A. L. Bertozzi, “Deep learning for real time crime forecasting,” arXiv preprint arXiv:1707.03340, 2017.
[28] X. Ouyang, C. Zhang, P. Zhou, H. Jiang, and S. Gong, “Deeppace: An online deep learning framework for mobile big data to understand human mobility patterns,” arXiv preprint arXiv:1610.07009, 2016.
[29] D. Yao, C. Zhang, Z. Zhu, Q. Hu, Z. Wang, J. Huang, and J. Bi, “Learning deep representation for trajectory clustering,” Expert Systems, vol. 35, no. 2, p. e12252, 2018.
[30] R. Jiang, X. Song, Z. Fan, T. Xia, Q. Chen, Q. Chen, and R. Shibasaki, “Deep roi-based modeling for urban human mobility prediction,” Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 4, no. 3, p. e12252, 2020.
proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, vol. 2, no. 1, pp. 1–29, 2018.

F. Xiong, X. Shi, and D.-Y. Yeung, “Spatiotemporal modeling for crowd counting in videos,” in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5151–5159.

H. Yao, F. Wu, J. Ke, X. Tang, Y. Jia, S. Lu, P. Gong, J. Ye, and Z. Li, “Deep multi-view spatial-temporal network for taxi demand prediction,” in Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

J. Zhang, Y. Zheng, and D. Qi, “Deep spatio-temporal residual networks for citywide crowd flows prediction,” in Thirty-First AAAI Conference on Artificial Intelligence, 2017.

H. Wei, H. Zhou, J. Sankaranarayanan, S. Sengupta, and H. Samet, “Residual convolutional lstm for tweet count prediction,” in Companion Proceedings of The Web Conference 2018. International World Wide Web Conferences Steering Committee, 2018, pp. 1309–1316.

E. Walach and L. Wolf, “Learning to count with cnn boosting,” in European Conference on Computer Vision. Springer, 2016, pp. 660–676.

R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. López-Sastre, S. Maldonado-Bascón, and D. Onoro-Rubio, “Extremely overlapping vehicle counting,” in Iberian Conference on Pattern Recognition and Image Analysis. Springer, 2015, pp. 423–431.

C. Arteta, V. Lempitsky, and A. Zisserman, “Counting in the wild,” in European conference on computer vision. Springer, 2016, pp. 483–498.

T. Zhang, D. Hazarika, S. Poria, and E. Cambria, “Recent trends in deep learning based natural language processing,” IEEE Computational Intelligence Magazine, vol. 13, no. 3, pp. 55–75, 2018.

Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” nature, vol. 521, no. 7553, pp. 436–444, 2015.

A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, “Deep learning for computer vision: A brief review,” Computational Intelligence and Neuroscience, vol. 2018, 2018.

A. Ioannidou, E. Chatziari, S. Nikolopoulos, and I. Kompatsiaris, “Deep learning advances in computer vision with 3d data: A survey,” ACM Computing Surveys (CSUR), vol. 50, no. 2, p. 20, 2017.

Q. Yuan, W. Zhang, C. Zhang, X. Geng, G. Cong, and J. Han, “Pred: periodic region detection for mobility modeling of social media users,” in Proceedings of the Tenth ACM International Conference on Search and Data Mining. ACM, 2017, pp. 263–272.

R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen, “Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6856–6864.

C. Li, P. Wang, S. Wang, Y. Hou, and W. Li, “Skeleton-based action recognition using lstm and cnn,” in 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). IEEE, 2017, pp. 585–590.

S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, “Convolutional lstm network: A machine learning approach for precipitation nowcasting,” in Advances in neural information processing systems, 2015, pp. 802–810.

A. Stec and D. Klabjan, “Forecasting crime with deep learning,” arXiv preprint arXiv:1806.01486, 2018.

Z. Yuan, X. Zhou, and T. 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. ACM, 2018, pp. 984–992.

Y. Liu, H. Zheng, X. Feng, and Z. Chen, “Short-term traffic flow prediction with conv- lstm,” in 2017 9th International Conference on Wireless Communications and Signal Processing (WCSP). IEEE, 2017, pp. 1–6.

C. Zhang, H. Li, X. Wang, and X. Yang, “Cross-scene crowd counting via deep convolutional neural networks,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 833–841.

B. Liao, J. Zhang, M. Cai, S. Tang, Y. Gao, C. Wu, S. Yang, W. Zhu, Y. Guo, and F. Wu, “Dest-r-net: A deep spatiotemporal residual network for hotspot traffic speed prediction,” in Proceedings of the 26th ACM international conference on Multimedia, 2018, pp. 1883–1891.

A. F. Agarap, “Deep learning using rectified linear units (relu),” arXiv preprint arXiv:1803.08375, 2018.

M. M. Lau and K. H. Lim, “Investigation of activation functions in deep belief network,” in 2017 2nd international conference on control and robotics engineering (ICCRE). IEEE, 2017, pp. 201–206.

G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” arXiv preprint arXiv:1207.0580, 2012.