In *Proceedings of the 9th International Workshop on Modeling Social Media (MSM'2018) Applying Machine Learning and AI for Modeling Social Media* , pages 1309-1316, Lyon, France, April 2018.

Residual Convolutional LSTM for Tweet Count Prediction

Hong Wei*
Department of Computer Science
University of Maryland
College Park, Maryland
hyw@cs.umd.edu

Hao Zhou
Department of Computer Science
University of Maryland
College Park, Maryland
hzhou@cs.umd.edu

Jagan Sankaranarayanan
UMIACS
University of Maryland
College Park, Maryland
jagan@umiacs.umd.edu

Sudipta Sengupta
Amazon Web Services (AWS)
Seattle, Washington
sudipta@amazon.com

Hanan Samet
Department of Computer Science
University of Maryland
College Park, Maryland
hjs@cs.umd.edu

ABSTRACT

The tweet count prediction of a local spatial region is to forecast the number of tweets that are likely to be posted from that area over a relatively short period of time. It has many applications such as human mobility analysis, traffic planning, and abnormal event detection. In this paper, we formulate tweet count prediction as a spatiotemporal sequence forecasting problem and design an end-to-end convolutional LSTM based network with skip connection for this problem. Such a model enables us to exploit the unique properties of spatiotemporal data, consisting of not only the temporal characteristics such as temporal closeness, period and trend properties, but also spatial dependencies. Our experiments on the city of Seattle, WA as well as a larger city of New York City show that the proposed method consistently outperforms the competitive baseline approaches.

CCS CONCEPTS

• **Applied computing**; • **Networks → Social media networks**;
• **Computing methodologies → Neural networks**;

KEYWORDS

Social Network, Twitter, Tweet Count Prediction, LSTM, Convolution, Convolutional LSTM, Residual Neural Network

ACM Reference Format:
Hong Wei, Hao Zhou, Jagan Sankaranarayanan, Sudipta Sengupta, and Hanan Samet. 2018. Residual Convolutional LSTM for Tweet Count Prediction. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, Article 4, 8 pages. https://doi.org/10.1145/3184558.3191571

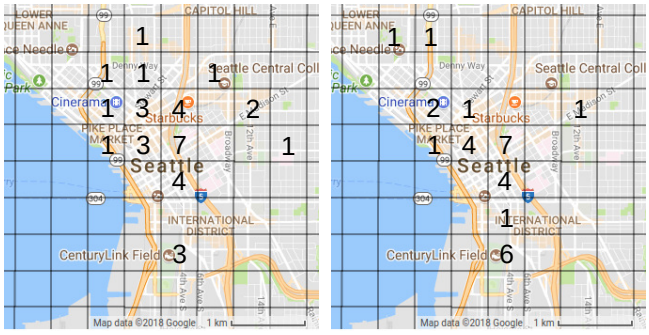
*This work is partially supported by Microsoft Research.

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3191571

1 INTRODUCTION

Given a geographical region (e.g., New York City), the goal of tweet count prediction is to forecast the spatial distribution of number of tweets that are likely to appear in the next time frame based on the previously observed data. Such a problem has many applications such as human mobility modeling [31] and abnormal event detection [3, 13, 15]. Taking abnormal event detection as an example, one may compare the predicted tweet count with the actual number of tweets in a geospatial local region. A significant difference is considered as a strong indicator of the occurrence of an abnormal event.



(a) (b)

Figure 1: (a) Tweet Count Distribution around the Seattle city center at 17:00-17:30 on 2016-07-16. (b) Tweet Count Distribution around around the Seattle city center at 17:30-18:00 on 2016-07-16. (A number in a grid cell refers to the value of tweet count at that time interval while an empty grid cell means no tweets.)

It is, however, challenging to make high-quality predictions of tweet count in a region due to both the spatial and temporal dependences. For example, Figure 1 gives the tweet count in two consecutive time interval around the Seattle city center area. The number in each grid cell refers to the value of tweet count at that time interval while an empty grid cell means no tweets. We notice that: (1) The number of tweets in a grid cell is positively correlated with that of nearby cells, i.e., a grid cell tends to have larger (smaller)

number of tweets if nearby cells also have larger (smaller) number of tweets, indicating the spatial dependences between cells. (2) The difference of the number of tweets between two temporal adjacent data is small, indicating the existence of temporal dependence. In fact, there are studies pointing out that spatiotemporal data also has a certain periodic pattern [13, 33], which indicates that we should also capture the periodic time-varying changes in tweet volume.

In this paper, we design an end-to-end model to predict the spatiotemporal tweet count sequence. Convolutional neural networks (CNNs) are designed to account for the spatial dependences of data. Zhang et al. [32] extend CNNs to account for temporal dependences by stacking spatial data of several consecutive time frames as input to CNNs, i.e., they simply treat spatial data at different time intervals as different channels of the input data. As a result, the way of encoding the temporal dependences is the same as that of spatial dependences, which may not be optimal. In this paper, we propose to apply the convolution LSTM (ConvLSTM) [24] layer as the basic stack unit which has convolutional structures in both the input-to-state and state-to-state transitions. In such a way, the spatial dependences are encoded by convolutional filters and temporal dependences are encoded by LSTM [10]. Both convolutional filters and LSTM play the role they are designed for. However, we notice that only using convolution LSTM cannot give us the best results. One reason may be that both convolutional neural network and LSTM are notorious for being highly non-convex and difficulty to converge to a good local minimum. Recent studies [16] have shown that using skip connections [6] can prevent the loss function from being chaotic, leading to a more convex loss function. Inspired by this and its effectiveness in many applications [6], we propose to add skip connections to our convolution LSTM. To further account for the temporal properties, we follow the idea of ST-ResNet [32] and partition sequences into 3 subsets: *closeness*, *period* and *trend* corresponding to recent, near and distant history, respectively. Each of these subsets of sequences is then separately fed into our method to generate an individual prediction which is then combined together to achieve the final prediction as discussed in [32]. We test the proposed method using two sets of geotagged tweets collected for Seattle, WA and New York City. Our experimental results demonstrate that the proposed method consistently outperforms the competitive baseline approaches.

To reiterate, the contributions of this paper are threefold. First, we are the first to apply ConvLSTM to tweet count problem, in which both convolutional filters and LSTM play the role they are designed for. Second, we add skip connections to ConvLSTM, which leads to a more convex loss function. It eases for the training procedure to find a better local minimum. Third, the proposed method achieved state-of-the-art results on two sets of geotagged tweets collected for Seattle, WA and New York city, showing the effectiveness of the proposed method.

2 RELATED WORK

As time goes by, the tweet counts in a region may be formulated as time series data, which enables the exploitation of the techniques like historical average and autoregressive integrated moving average (ARIMA) [9]. For example, TwitInfo [21] uses the weighted average of historical tweet counts to compute the expected frequency of tweets. Lin et al. [20] proposed a space-time autoregressive integrated moving average (STARIMA) model to predict urban traffic flow volume. Moreover, Chae et al. [3] adopt a similar model to seasonal ARIMA and decomposes the time series into the sum of a seasonal part, a trend part, and a remainder part, to check whether there exists an unusual volume of tweets.

Time series analysis based techniques, however, often neglect the effects exerted by nearby geographical regions when making predictions on a specific local region. Therefore, in their work on finding anomalies, Krumm and Horvitz [13] build a gradient boosting regression function that estimates the number of tweets on a region based on a list of features including the time of the day, the day of the week, and the tweet counts from neighboring regions.

With the recent advances in deep learning, a few recent studies have focused on introducing deep neural networks into modeling spatiotemporal data [2, 26]. For example, Shi et al. [24] propose a novel *convolutional LSTM* (*ConvLSTM*) network for precipitation nowcasting on radar echo spatiotemporal data, which enables the capture of both spatial and temporal correlation simultaneously by combining a convolution network and a recurrent LSTM network. Such a combination is done by innovatively replacing the matrix multiplication operations used in LSTM with convolution operations. This is different from Spatiotemporal Recurrent Convolutional Networks (SRCN) proposed in [30] which simply stack additional LSTM layers after convolutional layers.

Focusing on citywide crowd prediction, Zhang et al. [33] first partition historical spatiotemporal sequences into three subsets *closeness*, *period* and *trend*, which correspond to recent, near and distant history. Each subset is then fed into a Deep Convolution Neural Network to yield a prediction, and such predictions are then fused together along with external features such as week-of-day to produce the final forecast. Moreover, their subsequent work [32] further introduces the residual network [6] to capture citywide spatial dependence and gives better accuracy. Our method is different from them in the sense that we utilize ConvLSTM layers instead of regular convolution layers to build up our model, which shows effectiveness in our dataset.

3 METHOD

In this section, we first define the tweet count prediction problem in Section 3.1. Next, we briefly review a few key technologies used in our model such as Convolutional LSTM (ConvLSTM) [24] (Section 3.2), Deep Residual Network [7] (Section 3.2), and temporal property fusion [32] (Section 3.4). Finally, we present the design of our model in Section 3.5.

3.1 Tweet Count Prediction Problem

The goal of tweet count prediction is to use previously observed historical tweet count data in a local region to forecast the number of tweets in the next time step. In practice, a region can be represented by an $M \times N$ grid map based on the longitude and latitude. Thus, the observation at time step t can be represented by a tensor $\mathbf{X_t} \in \mathbb{R}^{M \times N}$ where $\mathbf{X_t}(m, n)$ is the tweet count in the grid cell (m, n) at time step t. Therefore, the tweet count prediction problem is formulated as follows:

Definition 3.1. The tweet count prediction problem $\mathcal{P}$ is to generate a prediction $\mathbf{Y}_T$, which is an estimation of $\mathbf{X}_T$, given a list of historical observations $\{\mathbf{X}_t | t = 0, \cdots, T-1\}$.

3.2 Convolutional LSTM

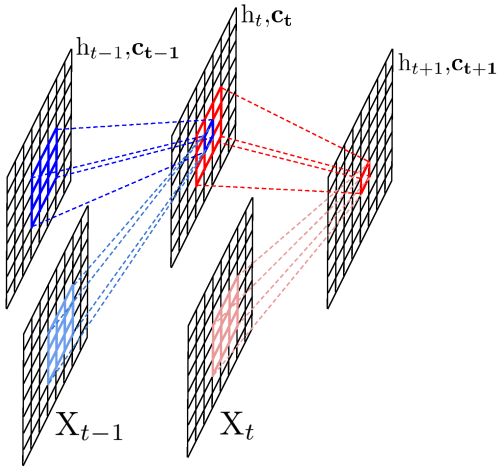


Figure 2: The Inner Structure of ConvLSTM. The LSTM matrix multiplication is replaced with convolution.

The Long Short-Term Memory (LSTM) network, one of the well-known recurrent neural networks, has achieved great success in many applications such as sequence modeling and especially sequence prediction [5, 11, 25]. Despite its strong ability in modeling temporal dependences of sequences, LSTM ignores spatial information when the sequence data is multi-dimensional. To overcome this drawback, Shi et al. [24] proposed the Convolutional LSTM (ConvLSTM) which innovatively uses a convolution operator in the state-to-state and input-to-state transitions (see Figure 2). The key equations in ConvLSTM are shown as follows:

$$\begin{aligned}
\mathbf{i_t} &= \sigma(\mathbf{W}_{xi} * \mathbf{X}_t + \mathbf{W}_{hi} * \mathbf{h}_{t-1} + \mathbf{W}_{ci} \circ \mathbf{c}_{t-1} + \mathbf{b}_i) \\
\mathbf{f_t} &= \sigma(\mathbf{W}_{xf} * \mathbf{X}_t + \mathbf{W}_{hf} * \mathbf{h}_{t-1} + \mathbf{W}_{cf} \circ \mathbf{c}_{t-1} + \mathbf{b}_f) \\
\mathbf{c_t} &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ tanh(\mathbf{W}_{xc} * \mathbf{X}_t + \mathbf{W}_{hc} * \mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{o_t} &= \sigma(\mathbf{W}_{xo} * \mathbf{X}_t + \mathbf{W}_{ho} * \mathbf{h}_{t-1} + \mathbf{W}_{co} \circ \mathbf{c}_t + \mathbf{b}_o) \\
\mathbf{h_t} &= \mathbf{o}_t \circ tanh(\mathbf{c}_t)
\end{aligned} \tag{1}$$

where t iterates from 1 to $T-1$. The variables $\mathbf{X}_t$, $\mathbf{c}_t$, $\mathbf{h}_t$, $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ are tensors to represent values of the inputs, cell outputs, hidden states, input gates, forget gates and output gates. σ is a logistic sigmoid function. The operator $\circ$ denotes the Hadamard product, i.e., element-wise product of matrix. And $*$ denotes the convolution operator instead of matrix multiplication, which is a key difference from FC-LSTM [5]. At last, $\mathbf{W}_*$ and $\mathbf{b}_*$ are weight and bias matrices parameters which need to be learned during training.

3.3 Residual Network

It is well known that deeper networks can model more complex functions and thus are more expressive. However, networks that work well in practice usually cannot be very deep. This is due to the vanishing gradient problem. To avoid this vanishing gradient problem and make the design of a deeper network possible, [6] proposed skip connections which directly link the output of lower layers to the input of higher layers. This shortcut has proven to be effective to alleviate the vanishing gradient problem in the training process and achieved significantly better performance in many applications. Recently, [16] has shown that skip connections can also help to prevent the loss function from being chaotic, leading to a more convex loss function, and thus, making it easy to find a good local minimum. Essentially, a residual building block can be defined as:

$$\mathbf{Y} = \mathcal{F}(\mathbf{X}) + \mathbf{X}, \tag{2}$$

where $\mathbf{X}$ and $\mathbf{Y}$ are the input and output tensors of the residual block. The function $\mathcal{F}$ represents several convolutional or ConvLSTM layers [8, 32, 34]. In this study, we always use the ConvLSTM [24] to assemble the residual block, which is illustrated in Figure 3. This is a key difference from ST-ResNet [32] which uses a regular convolutional layer instead as shown in Figure 3.

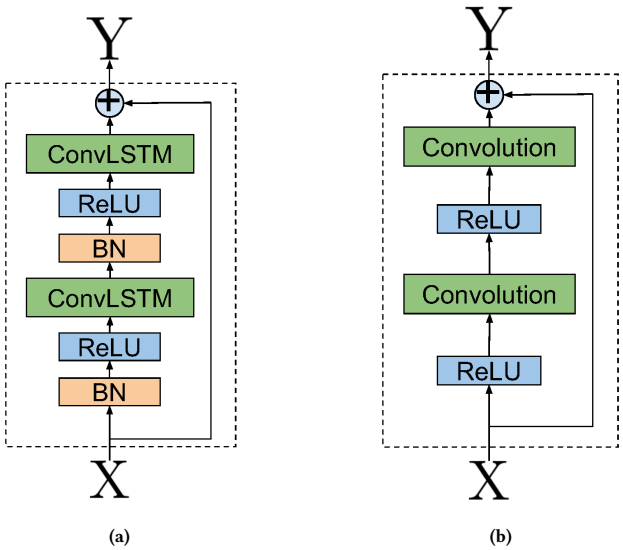


Figure 3: (a) Residual ConvLSTM block. (b) Residual block in ST-ResNet. BN: Batch Normalization

3.4 Temporal Properties Fusion

Zhang et al. [32, 33] pointed out that in spatiotemporal data sequences, making predictions on the future observations does not only rely on the observations of *recent time* but also depends on those in *near history* and *distant history*. Such temporal dependencies are modeled as temporal *closeness*, *period* and *trend*. More specifically, the temporal *closeness* dependence sequence is a l_c-long list of consecutive observations before the current time step and can be denoted by $X_t^c = \begin{bmatrix} X_{t-l_c} & X_{t-(l_c-1)} & \cdots & X_{t-1} \end{bmatrix}$. The temporal *period* dependence sequence is a l_p-long list of historical observations which are periodically chosen with a time interval p:

$X_t^p = \left[X_{t-p\cdot l_p} \quad X_{t-p\cdot(l_p-1)} \quad \cdots \quad X_{t-p\cdot 1}\right]$. Similarly, the temporal *trend* dependence sequence is a l_q-long list of historical observations which are also periodically chosen but with time interval q: $X_t^q = \left[X_{t-q\cdot l_q} \quad X_{t-q\cdot(l_q-1)} \quad \cdots \quad X_{t-1\cdot q}\right]$. In practice, p is set to a period of one-day to capture daily periodicity and q is set to one-week to reveal weekly trend.

Each of X_t^c, X_t^p and X_t^q are separately fed into three designated neural networks, which have the same structure but different weights, to generate observation predictions Y_t^c, Y_t^p and Y_t^q, respectively. At last, a parametric-matrix-based fusion is adopted to combine the three outputs Y_t^c, Y_t^p and Y_t^q to yield the final prediction $\mathbf{Y_t}$ [32] using the following equation:

$$\mathbf{Y_t} = \mathbf{W^c} \circ \mathbf{Y_t^c} + \mathbf{W^p} \circ \mathbf{Y_t^p} + \mathbf{W^q} \circ \mathbf{Y_t^q} \tag{3}$$

where $\mathbf{W}^*$ are weight matrices that balance different components. Additionally, features such as the time of the day and the day of the week can also be incorporated into $\mathbf{Y_t}$ using fully-connected layers.

3.5 Building Our Model

In this section, we present our model used for tweet count prediction. The structure of our model is illustrated in Figure 4.

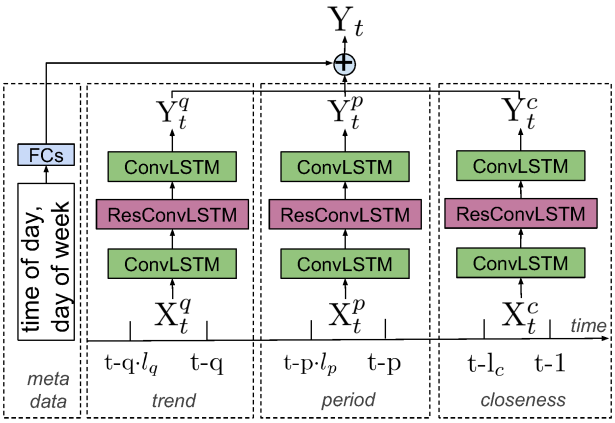
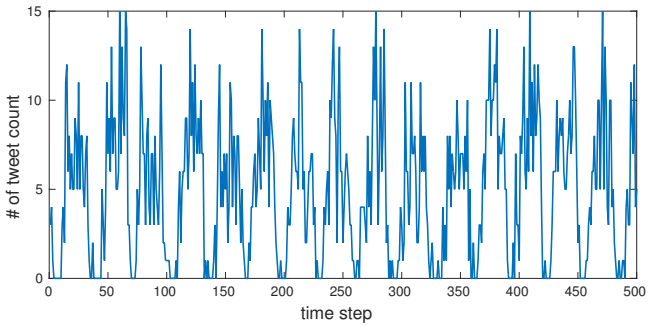
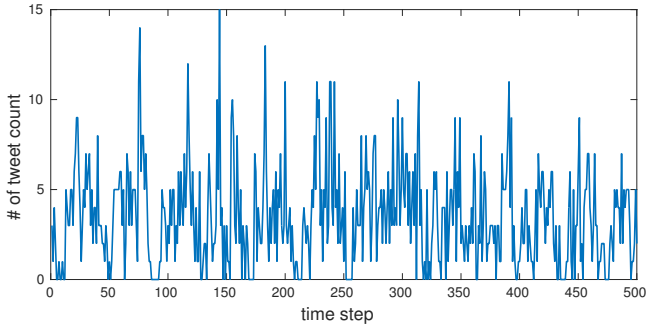


Figure 4: Our Model. ResConvLSTM: Residual ConvLSTM block; FCs: Fully-Connected Layers, i.e. Dense layers.

Similar to [32], we define our model to have three branches: *closeness*, *period* and *trend*, to incorporate periodic information in our data. This is because our data reveals positive correlation between adjacent time steps as well as periodic ones such as daily and weekly patterns. For example, Figure 5 draws the tweet counts in a region for 500 time steps in the city of Seattle and NYC, respectively. The two regions are the bold grid cells marked in Figure 7. The results show that our data indeed have certain temporal periodical pattern. As a result, in order to predict an expected tweet count Y_t at time step t, we break the historical observations to extract the *closeness*, *period* and *trend* dependence sequences X_t^c, X_t^p and X_t^q which are defined in Section 3.4. Each of the three dependence sequences is then fed into a designated network with the same structure but different weights to get the three predictions Y_t^c, Y_t^p and Y_t^q, respectively. These three predictions, together with meta data prediction, are combined using the parametric-matrix-based fusion to generate our final prediction as discussed in section 3.4. Please note that we can also define our model to have only one branch which takes a very long-range time series data so as to capture temporally periodical properties. However, this will introduce a huge amount of parameters, which is not only memory demanding, but also makes the networks much harder to train and slower to converge.



(a)



(b)

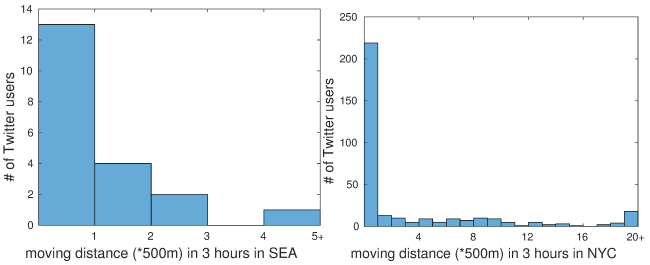
Figure 5: Temporal Pattern. (a) Seattle City; (b) NYC. Time step is in the unit of 30 minutes, starting from 18:30 on 2016-06-15.

As shown in Figure 4, each branch of our model has the same network structure, comprising of an input ConvLSTM layer, a ResConvLSTM block as described in Figure 3, and an output ConvLSTM layer. As a result of using ConvLSTM instead of convolutional layers as in [33] and [32], our model naturally takes a list of sequences as input and does not have to concatenate the long sequences e.g. X_t^c, X_t^p and X_t^q into one image-format-like tensor. Moreover, the outputs of the input ConvLSTM layer and ResConvLSTM block are in the form of a list of sequences which has the same length with the input such as X_t^c, X_t^p or X_t^q.

Except for the output ConvLSTM layer which has only 1 hidden state, all ConvLSTM layers are configured to have 32 hidden states. Since we only focus on predicting the expected spatiotemporal tweet count for the next time step, we set the output ConvLSTM layer to return one prediction sequence.

We define the size of the filter in our ConvLSTM to be 3×3. This is because the spatial correlation of tweet count data is quite local,

i.e., the number of tweets in a grid is correlated with the ones in the nearby grids instead of grids farther away. For example, Figure 6



(a) (b)

Figure 6: Histogram of Moving Distance of Twitter Users. We only consider Twitter users who have 2 or more geotagged tweets in the 3-hour time period starting from 18:30 on 2016-06-15. The moving distance of a user is calculated as the largest distance between the GPS coordinates in his geotagged tweets.

shows the histogram of moving distance of Twitter users during a time period of 3 hours in the city of Seattle and NYC, respectively. We notice that the majority of Twitter users travel less than 500 meters, i.e. less than the size of a grid cell.

Comparing with ST-ResNet [32], we replace its regular convolutional layers with ConvLSTM, as the latter is more powerful in capturing temporal dependence. Moreover, we stack only one residual block, instead of multiple blocks, because we empirically notice that adding more layers to our model cannot improve the performance of the model and sometimes results in over fitting. This also corresponds to the fact that Twitter users in our dataset usually have shorter moving distances.

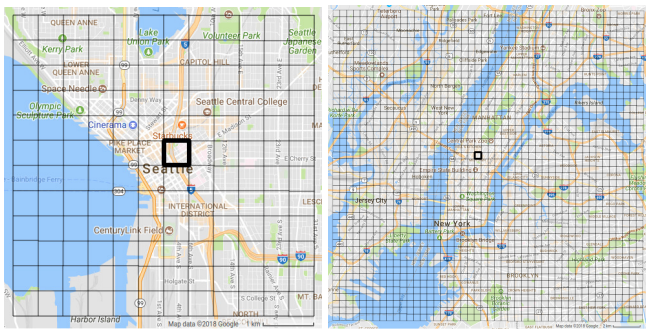
Meta-data features such as time-of-day, day-of-week are also incorporated in the model to capture the regular time-varying changes. To achieve this, we stack two fully-connected layers. The first is an embedding layer for features and the second maps from low to high dimensions to make the output have the same shape as the target [32].

4 EXPERIMENTS

All the experiments in this study are completed on an Nvidia GPU Quadro P6000 and the models are built using Keras [4] libraries with TensorFlow [1] as the backend.

4.1 Datasets

We use two sets of geotagged tweets collected from 2015-07-09 to 2017-09-30 in two cities: Seattle, WA (SEA) and New York City (NYC) to carry out all our experiments. The total number of tweets in each dataset is $1,025,181$ and $10,084,839$, respectively. Geotagged tweets are those that contain a pair of longitude and latitude coordinates values which indicate their location. These geotagged tweets are then aggregated into grid cells, which are $500m \times 500m$ squares spanning from [47.579784, -122.373135] to [47.633604, -122.293062] in SEA, and from [40.647984, -74.111093] to [40.853945, -73.837472] in NYC, which correspond to their metropolitan area, respectively. The two grid maps are shown in Figure 7, respectively. Note that the examples in Figure 1 and Figure 8 are illustrated on the inner 8×8 grid cells of Figure 7a as the boundary cells have few tweets to show. In this study, we define the interval of a time step to be 30 minutes, an empirical trade-off between the prediction promptness and accuracy. For example, the task of prediction prefers shorter temporal intervals as it gives more timely results. Shorter temporal intervals, however, might be too small to aggregate enough tweets for making high-quality prediction due to the sparsity of tweets.



(a) (b)

Figure 7: (a) 12×12 grid map in the Seattle. (b) 46×46 grid map in the NYC. The bold cells in each grid map are the chosen regions to draw Figure 5, respectively.

Removing Spam Tweets We identify two types of tweets as spam: (1) The tweets whose geographical coordinate values are the same as one of the city centers. Because such tweets are likely posted by accounts who simply give out a nominal location address (e.g., "Seattle, WA" and "New York City") which are then automatically geodecoded by the Twitter location service to city centers. Such accounts send out geo-targeted tweets spams such as "@tmj_sea_legal1" and they are very unlikely to be present exactly at the city centers. We removed $224,335$ and 0 tweets for Seattle and NYC in this step. (2) The tweets that are posted by suspicious Twitter users who behave more like bots, e.g., publishing more than 5 tweets at exactly the same location and 3 or more of such tweets are sent out only in 1 minute. We removed $204,800$ and $44,389$ tweets for Seattle and NYC datasets in this step. After filtering out spam tweets, we now have $756,457$ and $9,880,039$ tweets in the Seattle and NYC datasets, respectively.

Normalization The values of the tweet count are scaled to $[-1, 1]$ using Min-Max normalization [32]. Consequently, a tanh activation function is applied to the output for a faster convergence [14, 32]. To compare with the groundtruth, the predicted values are scaled back to normal ranges.

Training We split the data in each of the two cities into the training and the testing dataset, where the testing dataset contains the last 28 days of the observation sequences and the rest of the data belong to the training dataset. In so doing, we have $18,624$ training samples and $1,344$ testing samples for the city of Seattle, and $26,304$ training and $1,344$ testing samples for New York City.

The discrepancy between the numbers of training samples are due to occasional missing data on some days for each of the two cities. Following [32], our training procedure contains two steps. (1) To find a good initialization of our model, We first train our model using 90% of the training data and reserve the rest 10% as validation data. During this step, we apply early-stopping based on the validation loss. (2) After that, we continue to train our model on all the training data for another fixed number of epochs (e.g. 100 epochs). The loss function used in the training process is the Mean Squared Error.

By default, the periodicity and trend interval p and q are set to one day and one week, respectively. The lengths of the dependence sequences are set to $l_c = 3$, $l_p = 1$ and $l_q = 1$.

4.2 Baseline Approaches

We choose the following seven methods as the baseline approaches:

- *ZERO*: a naive baseline approach which simply yields predictions of 0s for all tweet count.
- *ARIMA*: Auto Regressive Integrated Moving Average (ARIMA) model is a time series analysis model for understanding the time series data or predicting future points in the series [9].
- *SARIMA*: Seasonal ARIMA, which additionally considers possible seasonal effects.
- *Eyewitness*: Eyewitness [13] uses gradient boosting regressors to train a regression function by considering features such as the time of the day, the day of the week and tweet counts from neighboring regions.
- *ST-ResNet*: ST-ResNet [32] is the currently state-of-the-art method used in spatiotemporal data prediction which is a strong baseline. Different from the proposed method, it uses regular convolution layers instead of convolutional LSTM layers. By default, ST-ResNet uses one residual block, which achieves the best results on our dataset. The effects of stacking multiple residual blocks will be further explored in Section 4.4.4.
- *ConvLSTM* $\times$ 3: a baseline approach that simply stacks three layers of ConvLSTM in order to contrast the effectiveness of a residual block over a ConvLSTM layer. It replaces the Residual ConvLSTM block with a ConvLSTM layer in Figure 4.
- *ConvLSTM* $\times$ 4: a baseline approach that stacks four layers of ConvLSTM in order to contrast the effectiveness of the skip connection in the residual block. We define this model by simply removing the skip connections in our proposed model.

4.3 Evaluation Metric

The results are measured by the Root Mean Square Error (RMSE):

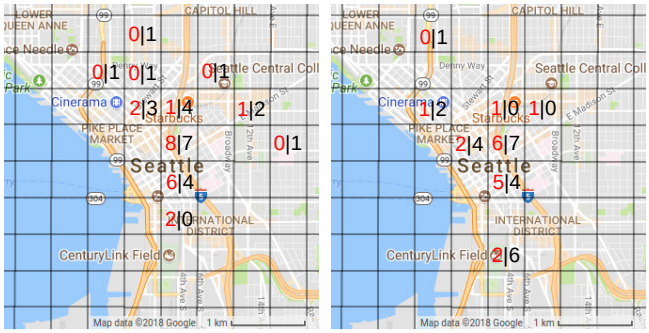
$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y_i - X_i)^2} \tag{4}$$

where n is number of testing cases, and Y_i and X_i are the prediction and groundtruth values, respectively.

4.4 Experimental Results

We start with an illustration of two predication examples, followed by a comparison between our proposed method and the six baselines mentioned in section 4.2. Then we study the effectiveness of temporal dependence sequences and the effect of deeper neural networks.

Figure 8 presents the prediction results using our model for the two tweet count distribution examples in Figure 1. The denotation in each grid cell is in the form of "prediction|groundtruth", referring to the prediction *vs.* groundtruth number of tweet count. The numbers in red are predictions. No denotation in a cell means a correct match with the groundtruth. The results show that both of the predications are generally good matches to the groundtruth by being able to capture the overall distribution of tweets as well as yielding only a slight difference for grid cells that have larger values of the tweet count. The error is mostly caused from predicting empty tweets for grid cells which have only one tweet. Such a situation is relatively arbitrary in the sense that the occurrence of such a tweet can be sporadic, which makes it hard to predict.



(a) (b)

Figure 8: (a) Prediction Example of Tweet Count Distribution around the Seattle city center at 17:00-17:30 on 2016-07-16. (b) Prediction Example of Tweet Count Distribution around around the Seattle city center at 17:30-18:00 on 2016-07-16. (The denotation in each grid cell is in the form of "prediction|groundtruth", referring to the prediction *vs.* groundtruth number of tweets. The numbers in red are predictions. No denotation in a cell means a correct match with the groundtruth.)

Table 1: Comparison Results (RMSE) on city of Seattle and NYC

Method	Seattle	NYC
ZERO	0.6353	1.2054
ARIMA	0.5117	0.5301
SARIMA	0.5242	0.5340
Eyewitness	0.4580	0.5332
ST-ResNet	0.4344	0.5166
ConvLSTM $\times$ 3	0.4659	0.5232
ConvLSTM $\times$ 4	0.4557	0.5278
Our Model	**0.4164**	**0.4879**

4.4.1 Compare with Baselines. Table 1 shows the results of seven baselines and the proposed method on two cities: Seattle and New York City. Simply generating prediction of 0s (ZERO) for every grid

cell performs much worse than all other methods. We notice that ST-ResNet outperforms all the other methods except the proposed one, showing its effectiveness. Using ConvLSTM achieves comparative results to ST-ResNet. We believe that this is because of the ability of ConvLSTM to model the spatial and temporal information well. The proposed method outperforms all the baselines and achieves state-of-the-art results. It achieves significantly better accuracies than both ConvLSTM × 3 and ConvLSTM × 4, which illustrates the effectiveness of the skip connections. As mentioned in [16], the loss functions of deeper networks are more likely to be chaotic, while adding skip connections can prevent this leading to a more convex loss function which is easier to train.

4.4.2 Effects of period and trend Dependence. We now investigate the performance of our model with and without utilizing *period* and *trend* information. We set the corresponding length variables l_q (l_q) to 0 or 1 to indicate whether the model is configured to use such information. The results are presented in Figure 9a. It shows that only using *closeness* information may perform even worse than the baselines and justifies the exploitation of *period* and *trend* dependence sequences. Nevertheless, in this study, we found that longer (> 2) *period* and *trend* dependence sequences do not always yield better accuracy.

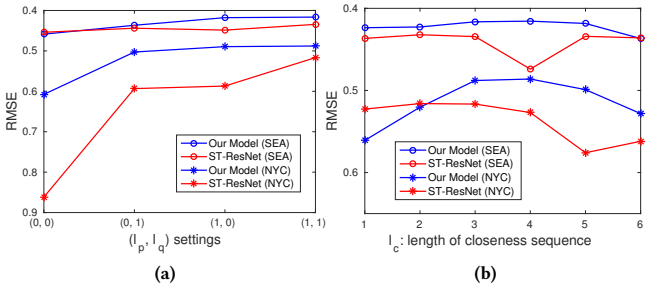


Figure 9: (a) Effects of using *period* and *trend* dependence or not. (b) Effects of length of *closeness* sequences. Note that the higher the curve, the smaller the RMSE value.

4.4.3 Effects of Length of closeness Dependence Sequences. In this subsection, we study whether a longer *closeness* dependence sequence can help achieve better performance in method ST-ResNet and in our model. The results are illustrated in Figure 9b. It can be seen that both models are able to achieve slightly better accuracy when the length begins to increase, but the performance saturates or becomes worse after l_c reaches 4. One possible reason is that the tweets that happened a longer time ago may not provide much information for predicting the tweet at the current time. Meanwhile, our model has higher gains than ST-ResNet because recurrent structure is more powerful in capturing temporal information. Moreover, we notice that ST-ResNet is more sensitive to tweets posted a longer time ago as the performance drops dramatically when $l_c = 4$ for Seattle and $l_c = 5$ for New York City.

4.4.4 Effects of Building Deeper Networks. In general, we found no significant gains by stacking more residual ConvLSTM blocks in our method ResConvLSTM. Take the city of Seattle for example, Figure 10 illustrates the results of stacking $\{0, 1, 2, 4\}$ residual blocks using RMSE metrics. It shows that two or more layers can not guarantee to achieve better results, although the performance deteriorates if no residual block is used at all. The situation is similar when it comes to stacking more residual convolutional blocks in baseline approach ST-ResNet. We believe this is due to the following two reasons: (1) As discussed in [16], deeper networks usually have a more chaotic loss function, making them difficult to train. (2) Deeper networks are more likely to suffer from over fitting.

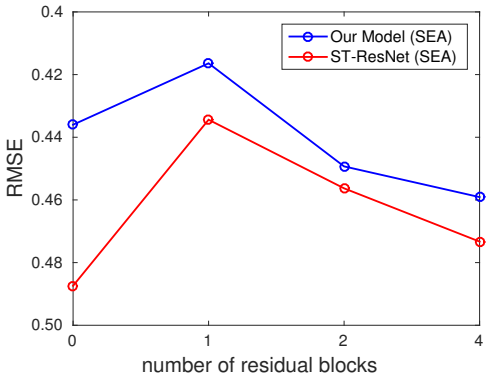


Figure 10: Results of stacking more residual blocks in the city of Seattle.

5 CONCLUSIONS

In this paper, we proposed a novel residual convolutional LSTM model for predicting tweet count. In essence, we utilize the framework of ST-ResNet [32] to model the temporal properties in spatiotemporal tweet count data such as *closeness*, *period* and *trend* dependence. To better capture the temporal correlation between sequences, we use ConvLSTM layers instead of regular convolution layers in ST-ResNet as the building block of the network. To make the network easier to train, we added skip connections. We evaluated the proposed method on the geotagged tweets collected for two cities: Seattle, WA and New York City. Our experiments show that the proposed method outperforms the baseline approaches and achieves state-of-the-art results. We carried out ablation studies and confirmed the necessity of utilizing the temporal properties *period* and *trend*. Finally, due to the fact that Twitter users have less intensive spatial moving activity, together with the data sparsity in some spatial area, we found that stacking more residual blocks to build deeper networks does not always yield better accuracy.

Predicting tweet count at a local place have many potential applications such as anomaly and event detection [13]. In the future, we will exploit our method on local news detection [12, 23, 28, 29]. The intuition is that if there is suddenly an abnormal change in the number of tweets at a location (like a significant increase), it probably means something is happening there. Specifically, one can first make a prediction on the number of tweets at a location to appear in the next time step. If the prediction is significantly less than the actual number of tweets, it might be considered as an anomaly, which likely corresponds to a local event.

In addition, instead of predicting the number of tweets at a location, we may also investigate the possibility of predicting the number of Twitter users at a location. This has many applications as

well such as population estimation and human mobility monitoring at city-wide scale.

Moreover, it is also interesting to extend the current model on tweets that don't have embedded GPS coordinates. We plan to approach this by applying geotagging procedures [17–19, 22, 27].

6 ACKNOWLEDGEMENT

We would like to thank Dr. John Krumm and Dr. Jin Li from Microsoft Research for providing supporting funding and the access to tweets of Twitter. This work was also supported in part by the National Science Foundation under Grant IIS-13-20791 and Grant CNS-1405688.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). https://www.tensorflow.org/ Software available from tensorflow.org.

[2] M. Ceci, R. Corizzo, F. Fumarola, D. Malerba, and A. Rashkovska. 2017. Predictive Modeling of PV Energy Production: How to Set Up the Learning Task for a Better Prediction? *IEEE Transactions on Industrial Informatics* 13, 3 (June 2017), 956–966. https://doi.org/10.1109/TII.2016.2604758

[3] J. Chae, D. Thom, H. Bosch, Y. Jang, R. Maciejewski, D. S. Ebert, and T. Ertl. 2012. Spatiotemporal social media analytics for abnormal event detection and examination using seasonal-trend decomposition. In *2012 IEEE Conference on Visual Analytics Science and Technology (VAST) (VAST '12)*. 143–152. https://doi.org/10.1109/VAST.2012.6400557

[4] Francois Chollet et al. 2015. Keras. https://github.com/fchollet/keras. (2015).

[5] Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. abs/1308.0850 (2013). arXiv:1308.0850 http://arxiv.org/abs/1308.0850

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), 770–778.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016 (CVPR '16)*. 770–778. https://doi.org/10.1109/CVPR.2016.90

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. *Identity Mappings in Deep Residual Networks.* Springer International Publishing, Cham, 630–645. https://doi.org/10.1007/978-3-319-46493-0_38

[9] S.L. Ho and M. Xie. 1998. The use of ARIMA models for reliability forecasting and analysis. *Computers & Industrial Engineering* 35, 1 (1998), 213 – 216.

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation.* 9, 8 (1997).

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[12] Alan Jackoway, Hanan Samet, and Jagan Sankaranarayanan. 2011. Identification of Live News Events Using Twitter. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks (LBSN '11)*. ACM, New York, NY, USA, 25–32. https://doi.org/10.1145/2063212.2063224

[13] John Krumm and Eric Horvitz. 2015. Eyewitness: Identifying Local Events via Space-time Signals in Twitter Feeds. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '15)*. ACM, New York, NY, USA, Article 20, 10 pages. https://doi.org/10.1145/2820783.2820801

[14] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. 1998. Efficient BackProp. In *Neural Networks: Tricks of the Trade.* Springer-Verlag, London, UK, UK, 9–50. http://dl.acm.org/citation.cfm?id=645754.668382

[15] Ryong Lee and Kazutoshi Sumiya. 2010. Measuring Geographical Regularities of Crowd Behaviors for Twitter-based Geo-social Event Detection. In *Proceedings of the 2Nd ACM SIGSPATIAL International Workshop on Location Based Social Networks (LBSN '10)*. ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/1867699.1867701

[16] H. Li, Z. Xu, G. Taylor, and T. Goldstein. 2017. Visualizing the Loss Landscape of Neural Nets. *ArXiv e-prints* (Dec. 2017).

[17] Michael D. Lieberman and Hanan Samet. 2011. Multifaceted Toponym Recognition for Streaming News. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '11)*. ACM, New York, NY, USA, 843–852. https://doi.org/10.1145/2009916.2010029

[18] Michael D. Lieberman and Hanan Samet. 2012. Adaptive Context Features for Toponym Resolution in Streaming News. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*. ACM, New York, NY, USA, 731–740. https://doi.org/10.1145/2348283.2348381

[19] Michael D. Lieberman, Hanan Samet, and Jagan Sankaranayananan. 2010. Geotagging: Using Proximity, Sibling, and Prominence Clues to Understand Comma Groups. In *Proceedings of the 6th Workshop on Geographic Information Retrieval (GIR '10)*. ACM, New York, NY, USA, Article 6, 8 pages. https://doi.org/10.1145/1722080.1722088

[20] Shu-Lan Lin, Hong-Qiong Huang, Da-Qi Zhu, and Tian-Zhen Wang. 2009. The application of space-time ARIMA model on traffic flow forecasting. In *2009 International Conference on Machine Learning and Cybernetics (ICMLC '09)*, Vol. 6. 3408–3412. https://doi.org/10.1109/ICMLC.2009.5212785

[21] Adam Marcus, Michael S. Bernstein, Osama Badar, David R. Karger, Samuel Madden, and Robert C. Miller. 2011. Twitinfo: Aggregating and Visualizing Microblogs for Event Exploration. In *CHI '11*. 227–236.

[22] Hanan Samet. 2014. Using Minimaps to Enable Toponym Resolution with an Effective 100% Rate of Recall. In *Proceedings of the 8th Workshop on Geographic Information Retrieval (GIR '14)*. ACM, New York, NY, USA, Article 9, 8 pages. https://doi.org/10.1145/2675354.2675698

[23] Jagan Sankaranarayanan, Hanan Samet, Benjamin E. Teitler, Michael D. Lieberman, and Jon Sperling. 2009. TwitterStand: News in Tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '09)*. ACM, New York, NY, USA, 42–51. https://doi.org/10.1145/1653771.1653781

[24] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada (NIPS '15)*. 802–810. http://papers.nips.cc/paper/5955-convolutional-lstm-

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS '14)*. MIT Press, Cambridge, MA, USA, 3104–3112. http://dl.acm.org/citation.cfm?id=2969033.2969173

[26] Akin Tascikaraoglu. 2018. Evaluation of spatio-temporal forecasting methods in various smart city applications. *Renewable and Sustainable Energy Reviews* 82 (2018), 424 – 435. https://doi.org/10.1016/j.rser.2017.09.078

[27] Faizan Wajid, Hong Wei, and Hanan Samet. 2017. Identifying Short-Names for Place Entities from Social Networks. In *Proceedings of the 1st ACM SIGSPATIAL Workshop on Recommendations for Location-based Services and Social Networks (LocalRec'17)*. ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3148150.3148157

[28] Hong Wei, Jagan Sankaranarayanan, and Hanan Samet. 2017. Finding and Tracking Local Twitter Users for News Detection. In *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL'17)*. ACM, New York, NY, USA, Article 64, 4 pages. https://doi.org/10.1145/3139958.3141797

[29] Hong Wei, Jagan Sankaranarayanan, and Hanan Samet. 2017. Measuring Spatial Influence of Twitter Users by Interactions. In *Proceedings of the 1st ACM SIGSPATIAL Workshop on Analytics for Local Events and News (LENS'17)*. ACM, New York, NY, USA, Article 2, 10 pages. https://doi.org/10.1145/3148044.3148046

[30] Haiyang Yu, Zhihai Wu, Shuqin Wang, Yunpeng Wang, and Xiaolei Ma. 2017. Spatiotemporal Recurrent Convolutional Networks for Traffic Prediction in Transportation Networks. In *Sensors*.

[31] Quan Yuan, Wei Zhang, Chao Zhang, Xinhe Geng, Gao Cong, and Jiawei Han. 2017. PRED: Periodic Region Detection for Mobility Modeling of Social Media Users. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM '17)*. ACM, New York, NY, USA, 263–272. https://doi.org/10.1145/3018661.3018680

[32] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. In *AAAI (AAAI '17)*.

[33] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. 2016. DNN-based Prediction Model for Spatio-temporal Data. In *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '16)*. ACM, New York, NY, USA, Article 92, 4 pages. https://doi.org/10.1145/2996913.2997016

[34] Yu Zhang, William Chan, and Navdeep Jaitly. 2017. Very deep convolutional networks for end-to-end speech recognition. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2017), 4845–4849.