Traffic Flow Prediction using Deep Spatio-Temporal Residual Networks

Arpith Reddy Singareddy
1217133827
asingar1@asu.edu

Manish Aakaram
1217852896
maakaram@asu.edu

Nishant Washisth
1217130460
nwashist@asu.edu

Sharad Saxena
1216924566
ssaxen18@asu.edu

Abhay Shrinivas Saraswathula
1217205626
asarasw2@asu.edu

CSE 575: Statistical Machine Learning
Computing, Informatics, and Decision Systems Engineering
Arizona State University

Under the guidance of
Prof. Yingzhen Yang

Abstract

In contemporary times, major metropolitan cities all around the world from New York to London, from Abu Dhabi to Beijing, all experience a significant amount of traffic on their roads. Considering the economic contribution and impact of these hubs, it becomes crucial to understand and manage the traffic flow of these cities. We are living in the era of Big Data and the Internet of Things, where sensors are recording nearly all aspects of our lives, and all activities leave a digital footprint. It is our goal to leverage this data in conjunction with state of the art machine learning techniques to develop a system that is capable of predicting the flow of traffic, and modelling the potential impact of unforeseen events such as storms, power outages, and pandemics.

1. Introduction

Accurate and timely traffic flow information is currently strongly needed for individual travellers, and government agencies [1]. It is important for managing the crowds during special events and incidents. Sudden massive crowds can lead to many issues such as stampedes. Examples of such incidents occur everywhere around the world. Stampedes are more evident in populous areas and are obvious during events such as Kumbh Mela in India, or Football Derbies or the Occurrence of augmented rare Pokémon when playing the game of 'Pokémon Go'. If such events involving high amounts of traffic flow can be predicted, one can make necessary adjustments to prevent dangerous situations from happening by issuing warnings, deploying more traffic control, or evacuating people in advance. Traffic flow prediction has also been a major player in the growth of Intelligent Transportation Systems (ITSs). There have been attempts to predict Traffic flow in the past but were not much successful due to the use of hand-designed features and lack of robust prediction models.

With the advent of IPv6 and the ubiquity of internet access, we are living in a world, where we have sensors to collect information about nearly every human activity and share it on the internet. This phenomenon, which is referred to as the Internet of Things[10] allows us to collect vast amounts of data about the activity that we are interested in, integrate it into our problem-solving apparatus and thereby improve our lives. A considerable amount of research is being done in the area of data mining from Big Data[11], to extract meaningful results and patterns from raw data. This avenue is a multidisciplinary endeavour where we have to take into account multiple factors such as volume, variety and velocity of data.

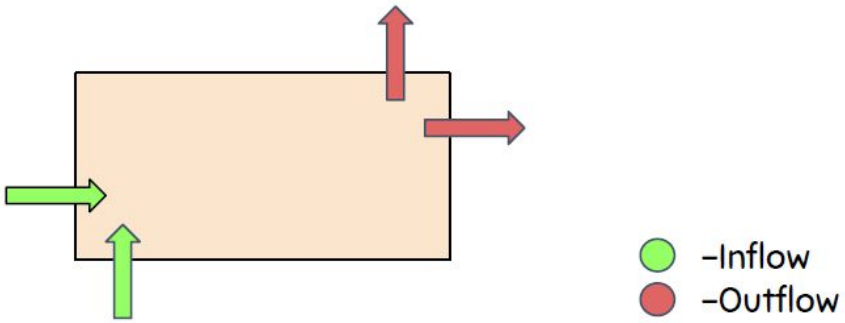

Fig. 1. Depiction of inflow and outflow in a region

Ultimately, the appropriate predictive method or descriptive method is applied to the data to achieve the desired results. In the case of our project, we will use GPS data from NYC taxi cabs to design our system.

When analyzing the data to predict the characteristics of traffic flow, we will need to define certain metrics that will be referenced throughout the project, so that we can get a clear picture of the situation. We will briefly describe some characteristics of our data. Inflow is a metric that describes the total traffic of crowds entering a region during a given time interval. Outflow denotes the total traffic of crowds leaving a region for other places during a given time interval. Figure 1 illustrates the concept.

In addition to these metrics, we must also consider that the data has spatial and temporal patterns that our model needs to capture. By spatial dependencies we mean the geographic position of one data point relative to another data point. Temporal dependencies refer to differences in data values with respect to time.

2. Problem Statement

The goal of the project can be defined as follows: Given historical observations of traffic flow at a given region for a set of sequential time intervals, the traffic flow for the next time interval must be predicted. External factors such as weather changes, special events can also be provided as parameters to improve the accuracy of prediction of traffic flow. The generated model should be able to capture the different types of dependencies such as Spatial, Temporal, and effects of any external factors that determine the traffic flow. The traffic flow of a region can be considered to consist of Inflow and Outflow (Zhang et al. 2016). Inflow can be defined as the amount of traffic entering a region and Outflow as the amount of traffic leaving a region.

In this project, we implement an approach using Deep Spatio-Temporal Residual Network [2] to solve the problem of traffic flow prediction. Deep Spatio-Temporal Residual Network or ST-ResNet is a combination of a series of Convolutional Layers and Residual Units [3]. The ST-ResNet primarily attempts to capture the Spatial and Temporal closeness, period, and trend properties along with the effects of external influence.

3. Preliminary Approach

This section discusses the preliminary approach for solving the problem of Traffic flow prediction briefly. As mentioned previously ST-ResNet is being used to address our problem. As per the problem statement, the prediction model must be able to capture Spatial as well as temporal dependencies of the traffic. Expanding on this, spatial dependency can be explained with a simple example of vehicles travelling from one region to another. There is a dependency between the two spatial regions where the vehicle travels from and the region where the vehicle travels to. Regarding Temporal dependencies, one can say from general knowledge that the traffic on the streets during weekdays is higher during the morning and evening periods when people travel to work and back to home compared to other times. There is also a spatial aspect to this, a region which might contain a workplace or office will experience higher traffic during weekdays whereas a residential area during a similar time experiences minimum traffic. We continue to discuss the preliminary implementation details of the solution.

4. Data and Preprocessing

For our implementation, we need data that describes the inflow and outflow of a particular region along with the time-period. As a part of the solution, the entire data for a given region is divided into a grid-like structure and assigned to the grid cells. Each cell in the grid has details regarding the inflow and outflow of traffic. Therefore, the entire data for a given region or city can be represented as a 2-channel image with height and width the same as the grid. Hence, Convolutional Neural Networks can be used to capture different relations between the data.

In this project, we operate with the NYC taxicab data provided by NYC Taxi and Limousine Corporation (TLC) [4]. The below image shows the sample data set before processing:

```
CMT;2009-01-05 16:02:52;2009-01-05 16:18:43;1;4.5;(-73.991063999999994,40.727654000000001);;;(-73.945770999999993,40.777650000000001);Cash;13.9;0;;0;0;13.9
CMT;2009-01-05 12:15:06;2009-01-05 12:27:58;1;1.7;(-74.001677999999998,40.747300000000003);;;(-73.978956999999994,40.750394);Cash;8.5;0;;0;0;8.5
CMT;2009-01-05 07:49:57;2009-01-05 07:54:11;1;1;(-73.982456999999997,40.731475000000003);;;(-73.973011,40.743386999999998);Credit;4.5;0;;1;0;5.5
```

The data set contains information about different taxicab trips taken in the city of New York over a period of time. Each row describes a trip and it contains information such as date, start time, duration, end time, pickup and drop-off coordinates, mode of payment, bill amount etc of the trip. Certain details of the trip such as the pickup coordinates and drop-off coordinates, start time,

duration of the trip are relevant to our problem. These details were pre-processed and represented in the form of a grid to calculate the inflow and outflow of each cell on the grid based on the pickup and drop-off points. We created a 32 * 32 grid to represent the entire New York City. The coordinates -74.50 (longitude), 40.90 (latitude) were used for the top left corner of the grid and the coordinates -73.70 (longitude), 40.50 (latitude) were used for the bottom right corner of the grid. Using these end coordinates and taking the step size as (max - min)/32, the coordinates for each of the 1024 cells were calculated. We separated the data set into half-hour chunks and for each of the half-hour, we created two grids, one for the pick up coordinates and other for the drop-off coordinates. Then for each of the trips present in the data set, we calculated the corresponding cell in which the pick-up coordinate would fall into and similarly the cell in which the drop-off coordinate would fall into, and the value of that corresponding cell would be increased by 1. For example, let's assume that the pickup date and time is 2009-01-05 16:02:52 and the pickup coordinate is -73.991063, 40.727654. So, we calculate the corresponding half-hour pickup grid using the pick up date and time, and then the corresponding cell in the grid in which the pick up coordinates would fall into and increase the count of the cell by 1.

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,39,65,31,32,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,0,0,1,0,2,82,30,31,50,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,14,123,9,37,36,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,59,65,6,112,25,0,0,0,0,0,0,0,2,0,1,0,0,2,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,4,80,28,25,215,3,0,11,5,0,0,0,0,0,1,0,0,0,1,0,1,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,26,88,5,55,203,7,13,24,27,5,0,27,2,0,1,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,0,0,33,74,8,84,97,3,29,38,20,4,4,3,26,0,1,1,1,0,1,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,0,0,10,114,165,47,135,20,8,33,46,24,0,1,2,3,0,0,1,1,1,
0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,0,0,4,155,87,86,75,7,7,30,46,14,5,2,6,6,0,0,4,1,0,1,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,10,105,115,167,32,4,14,10,11,7,5,18,9,4,4,0,1,5,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0,81,264,143,98,6,12,20,4,10,13,15,21,3,2,4,0,0,0,1,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,89,116,195,61,7,13,1,8,35,9,13,9,4,3,3,0,1,1,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,0,1,123,122,184,18,4,3,2,5,10,6,5,4,9,4,4,0,0,0,0,2,0,
0,0,0,0,0,0,0,0,0,0,1,0,0,0,4,0,3,155,103,210,29,9,17,0,0,2,0,2,0,3,3,8,1,0,2,1,0,1,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,10,84,135,162,21,2,26,14,0,0,0,4,2,1,5,13,6,1,1,2,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,4,49,137,129,13,8,7,12,0,0,1,0,1,2,1,7,6,2,0,1,0,1,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0,36,54,64,38,5,42,39,21,0,1,1,1,0,1,0,0,2,10,1,1,1,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,79,55,7,3,6,37,37,20,1,2,2,2,1,1,0,0,3,5,2,2,0,2,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,58,77,0,0,3,3,7,20,6,5,3,1,1,0,0,0,0,1,0,1,1,1,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,19,1,16,9,0,2,5,13,11,5,3,2,1,1,0,0,1,1,0,2,0,0,0,0,
```

Fig. 2. Sample grid for pickup coordinates after preprocessing the data set

The total pre-processing time for the data set of 2.4GB was around 20 minutes on an intel core-i5 processor-powered workstation. As shown in figure 2, a sample grid obtained for pick-up coordinates after pre-processing of the data set.

5. Model Architecture

To capture all the temporal dependencies of closeness, trend, and period, ST-ResNet is composed of 3 similar components of a series of Convolutional layers and Residual units. The convolutional layers are very well known for handling the spatial dependencies and aspects of the image data. We make sure that the convolutional layers are deep enough to capture the relation of the entire grid data over the image to the output of a particular cell. Vanishing gradients is a problem that arises when the number of layers increases in the neural network. To address this, we can use the residual units which again consist of Convolutional layers paired with ReLU [5] activations.

5.1 ST-ResNet Architecture:

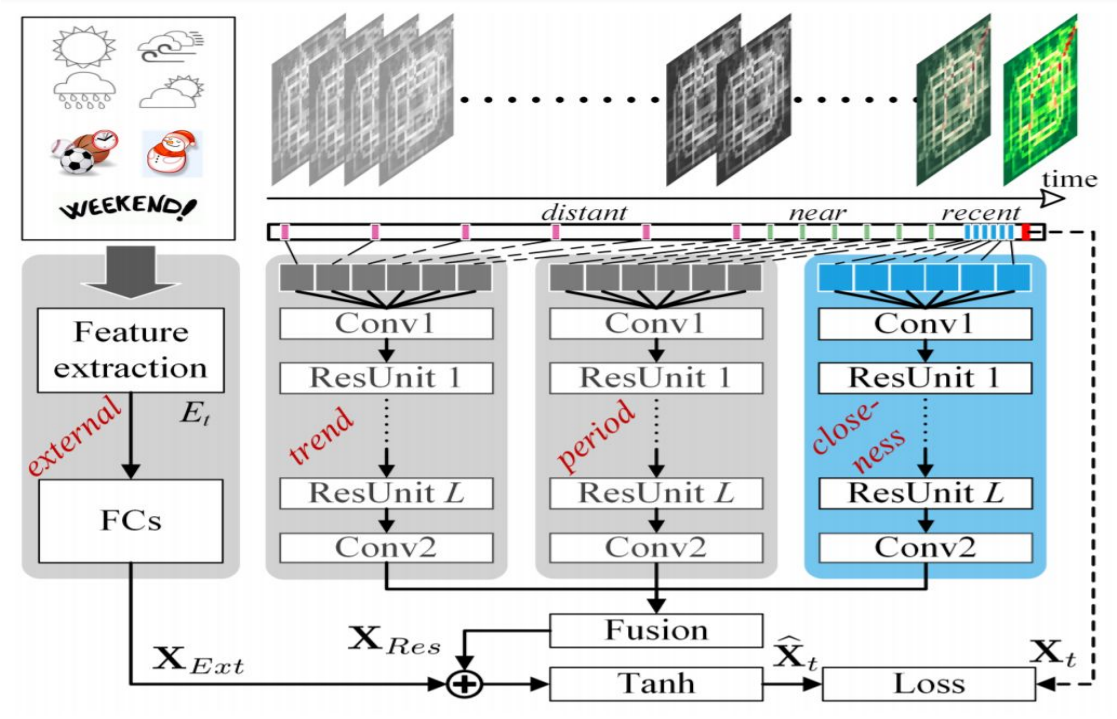

Fig. 3. ST-ResNet Architecture [2]

ST-ResNet is an End-to-End Deep learning model to capture the spatial and temporal dependencies. It consists of four major components to model for temporal closeness, period, trend, and external influence. As shown in figure 3, We create a 2-channel image-like matrix from inflow and outflow throughout a city at each time interval. The image like input data from preprocessing is used to select recent, near and distant time trajectories with a certain sequence length. This will generate a video like input data that is passed to each of the 3 components of the ST-ResNet. Each component will generate a 2 channel image after processing the video input through the residual network. The outputs from the 3 residual networks are fused together by additional layers with trainable weights to generate a single 2 channel image. The first three components share the same network structure with a convolutional neural network followed by a Residual Unit sequence. This fused output is then passed to the Tanh activation layer. The output from Tanh layer (range = [-1,1]) is rescaled back using min-max normalization values of training data. This output is a 2 channel image representing the inflow and outflow prediction values for the given input representing the past. During training, the loss is calculated using mean-squared error between predicted output and ground truth.

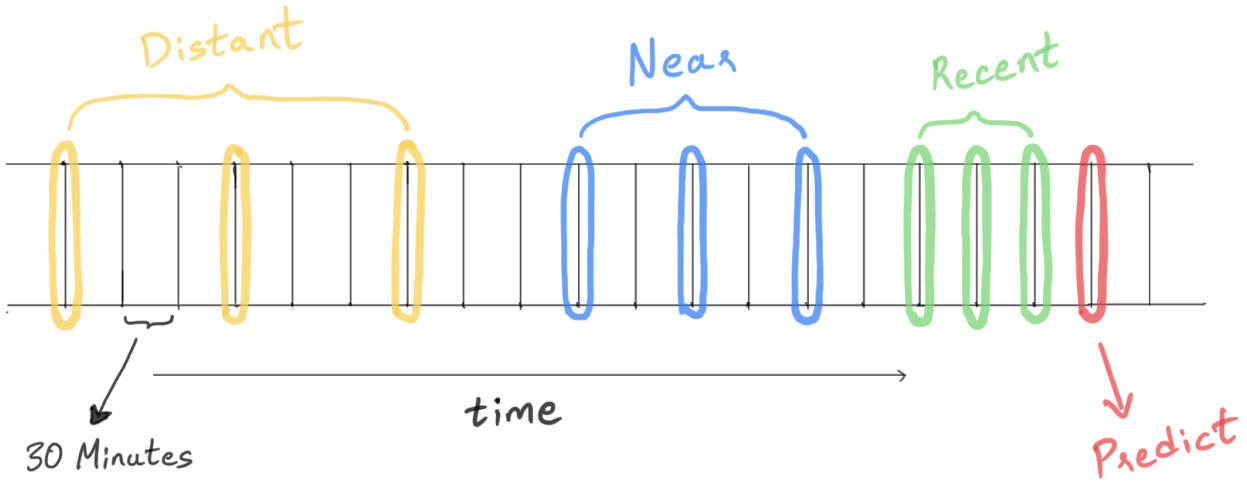

Fig. 4. Input Sequence and Prediction

5.2 Structure:

The first three components of ST-ResNet, temporal closeness, period and trend have the same network structure. Each component is composed of two sub-components: convolution and residual unit as shown in the figure below.

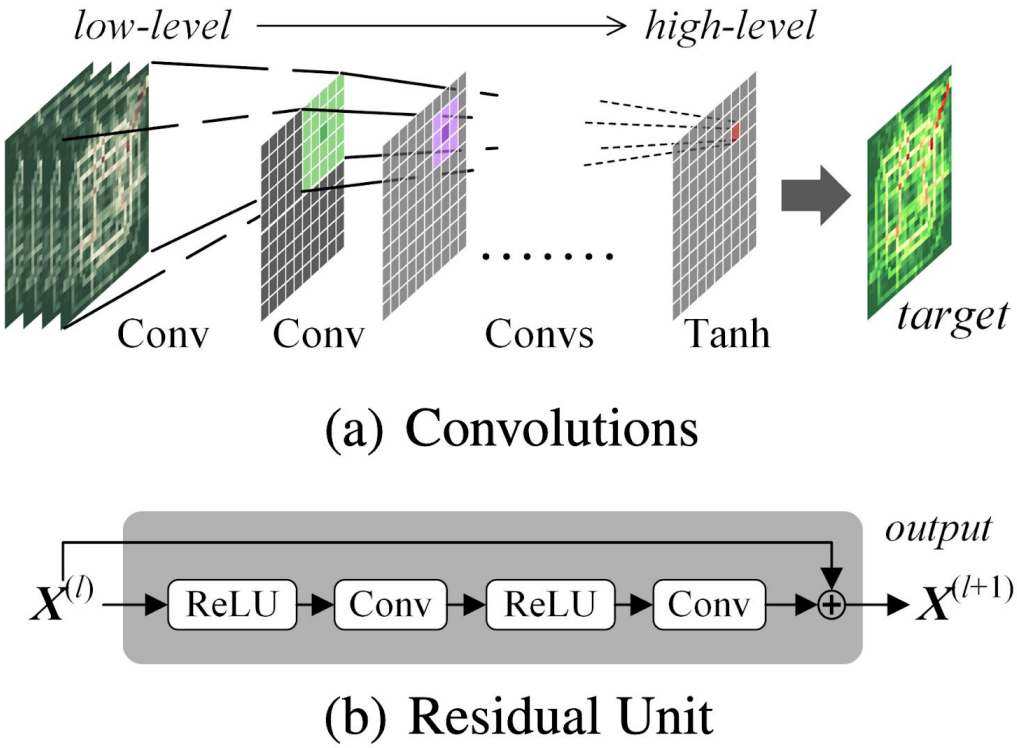

Fig. 5. Convolution and Residual Unit [2]

1. **Convolution:** A city is usually very large and consists of many regions with different sizes. The flow of nearby regions would affect each other. This can be effectively handled by the convolutional neural network(CNN). CNN is known to be very effective in capturing the spatial structural information hierarchically. CNN with many layers is required to capture spatial dependencies of any region as one convolution is limited by its kernel size and only accounts only for spatial near dependencies. We create three multiple levels of feature maps that are connected with convolutions as shown in figure 5(a). A node in the feature map of higher level depends on the nodes of the middle-level feature map, which depend on all nodes of the lower-level feature map/input. The *closeness* component in ST-ResNet models temporal closeness dependence by adopting a few 2-channel flows matrices of intervals in the recent time.

2. **Residual unit:** To capture very citywide dependencies, we would need a very deep convolutional network. If the kernel size of convolution is fixed to 3 × 3, and the input size is 32× 32, we would need more than 15 consecutive convolutional layers to model citywide dependencies. So, we use residual learning for training super deep neural networks over-1000 layers. L- residual networks are stacked on conv1 in ST-ResNet in figure 5(b). We add a convolutional layer(Conv2) on top of the Lth residual unit. We construct period and trend components similarly. The outputs from the convolutional layers of all the 3 components are then merged using parametric-matrix-based fusion which again contains learnable parameters. The inputs for all the 3 components comprise a sequence of images over time as mentioned previously. Before passing the input, all the sequences of images will be merged to form a single image with 2*sequence length channel image. The output generated will be a 2-channel image that depicts the inflow and outflow of the cells in the grid for the future time-period.
3. **External Component:** To incorporate external parameters, we extract weather data for the required time-period and this data is passed to fully connected layers to reduce the dimensions. The output is merged with the image produced from the previous step and then passed to a Tanh activation function which generates output for each cell in the range of [-1, 1]. Adam optimizer [6] is used to calculate the loss which is used to adjust the weights of the convolutional layer filters using backpropagation for a certain number of epochs.
4. **Fusion**: We use the 4 components in the ST-Resnet architecture for fusion. Initially, the trend, period and closeness components are fused based on a parametric-matrix approach and are combined with the remaining component i.e external component.

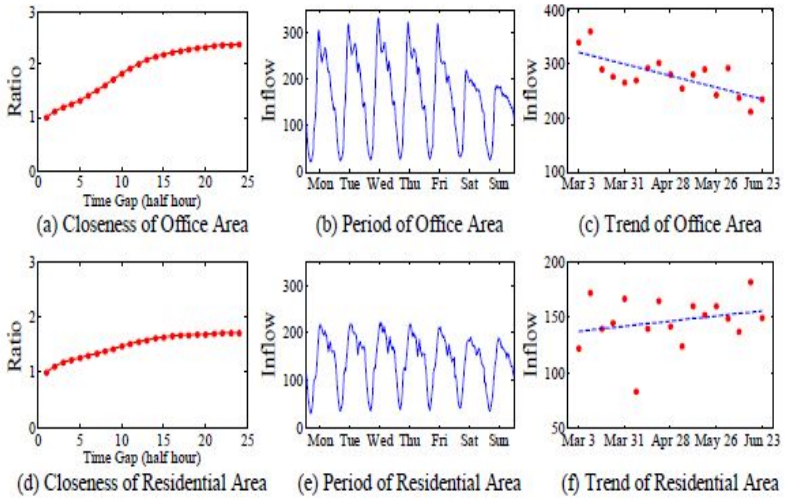

Fig. 6. Temporal dependencies [2]

Figure 6 represents the plot of the closeness, period and trend monitored for a week for two areas (Residential and Office). It can be identified that the closeness, period and trend have some effect on the inflow of traffic.

Closeness: A ratio of arbitrary inflows at every 30 minutes (i.e half an hour) interval. It can be observed that for Office areas, the inflow at the recent time gap has a high correlation unlike those at a distant time interval.

Period: It can be observed that for the weekdays in the office areas, the inflow is generally high, opposed to that in the weekends. For the residential areas, we have an almost constant trend.

Trend: The inflows are measured in the intervals 9:00 PM - 9:30 PM every Tuesday from March-June 2015. The inflow decreases in the office area over time and increases in the residential area.

We fuse the closeness, period and trend using the below formula:

$$\mathbf{X}_{Res} = \mathbf{W}_c \circ \mathbf{X}_c^{(L+2)} + \mathbf{W}_p \circ \mathbf{X}_p^{(L+2)} + \mathbf{W}_q \circ \mathbf{X}_q^{(L+2)}$$

o represents the Hadamard product and W_c , W_p and W_q represent the learnable parameters for closeness, period and trend.

We fuse the external component with the other components using the formula:

$$\hat{\mathbf{X}}_t = \tanh(\mathbf{X}_{Res} + \mathbf{X}_{Ext})$$

$\widehat{X}_t$ refers to the prediction at a time 't' and tanh makes sure our output is in the range [-1, 1]. We calculate the loss using the formula below. We train our ST-ResNet model such that the loss is minimized.

$$\mathcal{L}(\theta) = \|\mathbf{X}_t - \hat{\mathbf{X}}_t\|_2^2$$

θ refers to the learnable parameters in the model. The flow of implementation can be seen in figure 6.

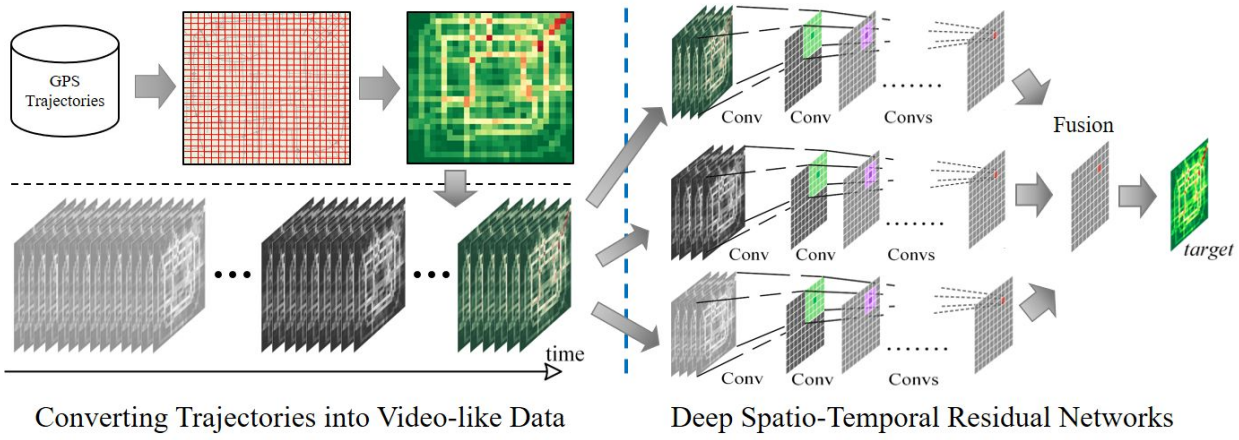

Fig. 6. Input Sequence and Prediction [12]

5.3 Algorithm and Optimization:

Algorithm 1: ST-ResNet Training Algorithm

Input: Historical observations: $\{\mathbf{X}_0, \cdots, \mathbf{X}_{n-1}\}$;
external features: $\{E_0, \cdots, E_{n-1}\}$;
lengths of *closeness*, *period*, *trend* sequences: l_c, l_p, l_q;
peroid: p; trend span: q.
Output: Learned ST-ResNet model
// construct training instances
$\mathcal{D} \longleftarrow \emptyset$
for *all available time interval* $t(1 \leq t \leq n-1)$ **do**
  $\mathcal{S}_c = [\mathbf{X}_{t-l_c}, \mathbf{X}_{t-(l_c-1)}, \cdots, \mathbf{X}_{t-1}]$
  $\mathcal{S}_p = [\mathbf{X}_{t-l_p \cdot p}, \mathbf{X}_{t-(l_p-1)\cdot p}, \cdots, \mathbf{X}_{t-p}]$
  $\mathcal{S}_q = [\mathbf{X}_{t-l_q \cdot q}, \mathbf{X}_{t-(l_q-1)\cdot q}, \cdots, \mathbf{X}_{t-q}]$
  // $\mathbf{X}_t$ is the target at time t
  put an training instance $(\{\mathcal{S}_c, \mathcal{S}_p, \mathcal{S}_q, E_t\}, \mathbf{X}_t)$ into $\mathcal{D}$
// train the model
initialize all learnable parameters θ in ST-ResNet
repeat
  randomly select a batch of instances $\mathcal{D}_b$ from $\mathcal{D}$
  find θ by minimizing the objective (6) with $\mathcal{D}_b$
until *stopping criteria is met*

Fig. 7. Algorithm and Optimization[2]

6. Training and Testing

We use the NYC taxi trip data [4] to perform training of the model and testing it. We use 80-20 train-test split mechanisms to create the training data and testing data. Since there are many training examples in the data, the loss calculation can be very slow leading to slow learning. To mitigate this, we are going to use Stochastic Gradient descent [7] where the entire train data will be split into batches randomly and during the training, backpropagation is done for each batch without waiting for the entire training data to be iterated. The training is done for 25 epochs where we start to see comparable training and test accuracy results.

7. Metrics

When the testing is done, we compare the results of our predicted model with that of the ground truth. Various metrics such as RMS error [9], Mean Absolute percentage error (MAPE) [8] can be calculated to get a better understanding of the performance of the trained model. We have also designed a custom accuracy metric which will give the accuracy percentage of a prediction in comparison with the ground truth. This will help us in better understanding the predictions made by the model.

8. Results

We have conducted experiments using ST-ResNet Model for predicting the Traffic flow of the New York Taxi trip dataset. This is done by varying the different hyper-parameters of the ST-ResNet Model. We have also compared the results of the existing work on the Spatio Temporal traffic flow prediction on the New York Taxi trip dataset.

The different hyperparameters that can be tuned in our implementation of the ST-ResNet model and the impact of each of them are described here. The sequence length is the count of images that are merged together to form a video-like input with a depth equal to twice of the sequence length. We have seen that as we increase the length of the sequence of the images considered for each input, the loss has consistently decreased indicating high prediction accuracy and lower error. This can be understood as greater the amount of information available to the current prediction, better is the result. However, this affects the efficiency of the system. Due to the increase in the size of the input it directly affects the performance of the model thereby increasing the runtime. Another important parameter to consider is the gap in between the different frames of the inputs to the trend, period and closeness component modules. It has been observed that increasing the gap in the trend period has improved the performance because it captures properties over a large period of time. Whereas the period component behaved in an oddly interesting way. As the gap is increased over a certain value, the mode's performance is increased until a certain value and then starts decreasing. The reason might be that the importance of the period component is no longer being captured because of the increase in the gap. It starts to model the same properties of the trend module which causes redundancy.

Regarding the width of the gap in the closeness component, we have observed that, closer the gap, better is the performance. This is similar to the way that the trend component is performing. We observed that increasing the number of images closer to the prediction timestamp, and also reducing the width of the gap highly increases.

We have compared our model's outputs with few of the works that were available to us on the New York Taxi trip dataset. The RMSE value that we have achieved with our implementation of ST-ResNet for a training setting of 25 epochs and a batch size 64 is around 7.69. These values are comparable with the results that were achieved by the authors for the original implementation in the paper[1]. There are also a few results that were obtained by different analysis using Random forest and Tree regressors. The results are better with the Deep neural network implementation compared the Random Forest and tree regressors. In the below table we compare our results with that of the others.

Model	RMSE
Random Boost and Tree Regressor	14.82
ST-ResNet (Junbo Zhang)	6.33
ST-ResNet (Ours)	7.69

In Figure 8, we show the pictures that have been predicted by our model and also a comparison with the ground truth. We can observe that the predicted image captures all the different nuances in the data and the images that are generated are very close to the ground truth values.

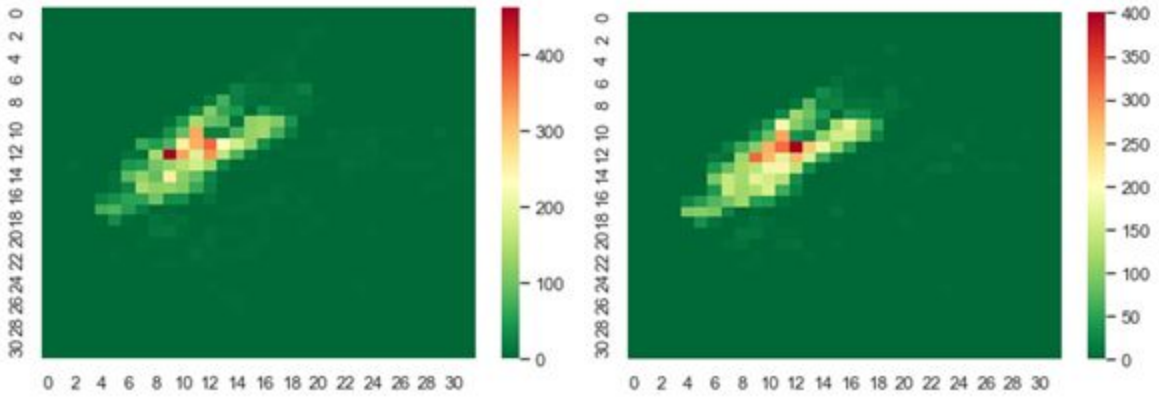

Figure 8(a) Predicted output; (b) Ground Truth

9. Future Work

Although the model worked pretty well, we see that it has drawbacks in modelling the temporal aspects of the traffic flow. The selection of time intervals and number of time intervals to consider are hyper parameters of the model which require a lot of tuning. To mitigate this, we are considering incorporating RNNs or LSTMs to better model the temporal aspects, thereby reducing the tuning and hopefully improving the performance. Recurrent neural networks and LSTM networks can be explored to handle the temporal dependencies with better performance.

10. References

[1] N. Zhang, F.-Y. Wang, F. Zhu, D. Zhao, and S. Tang, "DynaCAS: Computational experiments and decision support for ITS," IEEE Intell. Syst., vol. 23, no. 6, pp. 19–23, Nov./Dec. 2008.

[2] Junbo Zhang, Yu Zheng, Dekang Qi, "Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction" Jan 2017 https://arxiv.org/pdf/1610.00081.pdf

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[4] TLC Trip Record Data URL: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[5] Abien Fred Agarap, "Deep Learning using Rectified Linear Units (ReLU)" 2019 https://arxiv.org/abs/1803.08375

[6] Kingma, Diederik P. and Ba, Jimmy Adam: A Method for Stochastic Optimization. (2014). , cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015 .

[7] Ruder, S. (2016). An overview of gradient descent optimization algorithms. ArXiv Preprint ArXiv:1609.04747.

[8] de Myttenaere, B Golden, B Le Grand, F Rossi (2015). "Mean absolute percentage error for regression models", Neurocomputing 2016 arXiv:1605.02541

[9] Wikipedia contributors, "Root-mean-square deviation," *Wikipedia, The Free Encyclopedia,* https://en.wikipedia.org/w/index.php?title=Root-mean-square_deviation&oldid=941256353 (accessed September 30, 2020)

[10] Xia, Feng, et al. "Internet of things." International journal of communication systems 25.9 (2012): 1101.

[11] Singh, Singh. “Big Data-A Review.” i-Manager’s Journal on Information Technology 6.1 (2016): 25–. Print.

[12] https://www.microsoft.com/en-us/research/wp-content/uploads/2016/11/AAAI2017_overview.png