Group 55 - Reproducibility Project Blog

Reproduction results on the paper “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting” by Yaguang Li, Rose Yu, Cyrus Shahabi, Yan Liu.

Members and Contributions

Matias Bergerman (5626072 - M.Bergerman@student.tudelft.nl) - Reproduction of original code with added configurability.

Raphael Frühwirth (5897297 - r.fruehwirth@student.tudelft.nl) - Analysis of results and addition of visualizations.

Jin Young Choi (5896207 - jychoi99@gmail.com) - Evaluation of the algorithm with a new dataset.

Abstract

This blog post presents the results for a reproduction of the paper *“Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting”* by Yaguang Li, Rose Yu, Cyrus Shahabi and Yan Liu [1]. This project has been conducted during the CS4240 Deep Learning (2022/23 Q3) course. The code supplied by the authors of the paper has been adapted, and experiments were performed with a reduced number of traffic sensors, given the computation constraints. The algorithm has additionally been tested with a new dataset containing traffic information from the city of Tokyo [2]. The project repository with the code and figures is available at https://github.com/BigLob/DeepLearningProject.

Introduction

This project aims to reproduce the performance improvements that have been reported in [1] with respect to state-of-the-art baselines for traffic forecasting. The task that the algorithm attempts to accomplish is to predict the speed of traffic in a sensor network given the road network and past speed samples. The main challenges of this task are the complex spatial dependencies on the road network, non-linear temporal dynamics, and the inherent difficulty of long-term forecasting.

The main contribution of the paper’s approach lies in the representation of the data using a directed graph whose nodes are sensors and edge weights represent proximity between the sensor pairs measured by the road network distance. Furthermore, it proposes the *diffusion convolution* operation to capture the spatial dependency of traffic flow. This operation is integrated with the *sequence-to-sequence* architecture and *scheduled sampling* technique to obtain a Diffusion Convolutional Recurrent Neural Network (DCRNN).

The final architecture was reported to improve performance by up to 15% with respect to previous methods. On this project the DCRNN as well as baseline algorithms have been tested on the METR-LA dataset [3] which has been used on the paper, as well as the new EXPY-TKY dataset [2]. In all cases, the amount of network sensors as well as the number of timesteps was reduced to be able to perform the experiments within the computation constraints. For this reason, the values obtained for the prediction error are significantly higher, and performance can only be analyzed in relative terms to baseline methods. Nevertheless, we hope to obtain results which are meaningfully comparable to the ones reported on the paper, and in that way contribute to the reproducibility of Deep Learning research.

Methodology

The traffic forecasting task performed in this project is the prediction of future traffic speeds given previous traffic speed data from a set of traffic sensors distributed throughout the road network. We can construct a sensor network $G = (V, E, W)$ where V is the set of N sensors which are represented as vertices, E is a set of edges connecting the sensors, and $W \in \Re^{N \times N}$ is a weighted adjacency matrix containing the proximity of every sensor pair in terms of their arrangement in the road network.

Let's say that a traffic flow was observed on G as a graph signal $X \in \Re^{N \times P}$ where P is a number of features for each node. Also, we denote $X^{(t)}$as a graph signal observed on time t. Then the goal of the traffic forecasting problem is to learn a function $h(\cdot)$ that maps T' historical graph signals to future T graph signals given a graph G. This can be expressed as the following:

$$[\boldsymbol{X}^{(t-T'+1)}, \cdots, \boldsymbol{X}^{(t)}; \mathcal{G}] \xrightarrow{h(\cdot)} [\boldsymbol{X}^{(t+1)}, \cdots, \boldsymbol{X}^{(t+T)}]$$

The paper approaches this problem by modeling spatial and temporal dependencies of the traffic flow.

Spatial Dependency Modeling

Diffusion

The paper relates the traffic flow to a diffusion process. A diffusion process on a graph can be expressed as a random walk on G with restart probability alpha, and a state transition matrix $D_O^{-1} W D_0$ is a diagonal matrix, where W is the adjacency matrix of proximity, which has the number of out-degree on the diagonal. Then, after K random walks, or K state transitions, the likelihood of diffusion from a node to another node can be expressed as $(D_O^{-1} W)^K$. The distribution for the opposite direction of diffusion can be expressed as $(D_I^{-1} W^T)^K$.

Diffusion Convolution

We can define the diffusion convolution in terms of a bidirectional diffusion, which provides the model more flexibility for capturing the influence from both the upstream and downstream traffic. The diffusion convolution operation over a graph signal X and a filter f_θ is defined as following:

$$\boldsymbol{X}_{:,p} \star_{\mathcal{G}} f_{\boldsymbol{\theta}} = \sum_{k=0}^{K-1} \left(\theta_{k,1} \left(\boldsymbol{D}_O^{-1} \boldsymbol{W} \right)^k + \theta_{k,2} \left(\boldsymbol{D}_I^{-1} \boldsymbol{W}^\intercal \right)^k \right) \boldsymbol{X}_{:,p} \quad \text{for } p \in \{1, \cdots, P\}$$

The convolution is defined by graph signal $X \in \Re^{N \times P}$ multiplied by the summation of bidirectional transition matrices of diffusion $(D_O^{-1}W)^K$, $(D_I^{-1}W^T)^K$ weighted by $\theta \in \Re^{K \times 2}$ which are the parameters for the filter.

Diffusion Convolution Layer

The diffusion convolution layer maps P-dimensional features to Q-dimensional outputs for every node in the graph. The parameter tensor is defined as $\theta \in \Re^{Q \times P \times K \times 2}$, where $\theta_{q,p,:,:} \in \Re^{K \times 2}$ corresponds to the filter parameter of diffusion convolution mentioned above for the p-th input and the q-th output. The diffusion convolution layer can be written as follows:

$$\boldsymbol{H}_{:,q} = \boldsymbol{a} \left(\sum_{p=1}^{P} \boldsymbol{X}_{:,p} \star_{\mathcal{G}} f_{\boldsymbol{\Theta}_{q,p,:,:}} \right) \qquad \text{for } q \in \{1, \cdots, Q\}$$

Here, $X \in \Re^{N \times P}$ is the input and $H_{:,q} \in \Re^{N \times Q}$ is the output. $a(\cdot)$ is the activation function, such as ReLU or sigmoid.

Temporal Dynamics Modeling

To model temporal aspects of traffic data recurrent neural networks are employed, in particular Gated Recurrent Units (GRU). Matrix multiplications in GRU are substituted with the diffusion convolution, which leads to the final model in the paper: *Diffusion Convolutional Gated Recurrent Unit(DCGRU).*

$$\begin{aligned} \boldsymbol{r}^{(t)} &= \sigma(\boldsymbol{\Theta}_r \star_{\mathcal{G}} [\boldsymbol{X}^{(t)},\ \boldsymbol{H}^{(t-1)}] + \boldsymbol{b}_r) & \boldsymbol{u}^{(t)} &= \sigma(\boldsymbol{\Theta}_u \star_{\mathcal{G}} [\boldsymbol{X}^{(t)},\ \boldsymbol{H}^{(t-1)}] + \boldsymbol{b}_u) \\ \boldsymbol{C}^{(t)} &= \tanh(\boldsymbol{\Theta}_C \star_{\mathcal{G}} [\boldsymbol{X}^{(t)},\ (\boldsymbol{r}^{(t)} \odot \boldsymbol{H}^{(t-1)})] + \boldsymbol{b}_c) & \boldsymbol{H}^{(t)} &= \boldsymbol{u}^{(t)} \odot \boldsymbol{H}^{(t-1)} + (1 - \boldsymbol{u}^{(t)}) \odot \boldsymbol{C}^{(t)} \end{aligned}$$

In the equation, $X^{(t)}$ and $H^{(t)}$ correspond to the input and output at timestep t, $r^{(t)}$ and $u^{(t)}$ correspond to reset gate and update gate at time t, and $\star G$ denotes the diffusion convolution defined above.

The overall network architecture

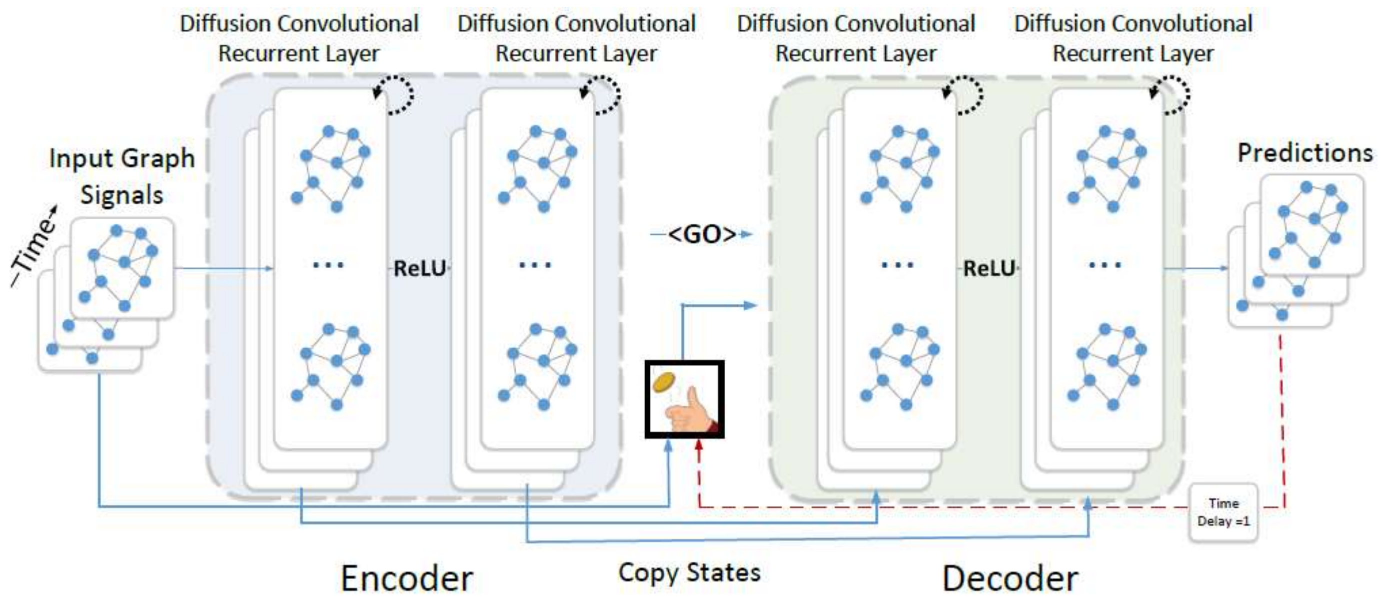

Diffusion Convolutional Recurrent Layer
Diffusion Convolutional Recurrent Layer
Diffusion Convolutional Recurrent Layer
Diffusion Convolutional Recurrent Layer
Input Graph Signals
Time
ReLU
<GO>
ReLU
Predictions
Time Delay =1
Encoder
Copy States
Decoder

Figure 1. The overall network architecture for spatiotemporal traffic forecasting. The encoder produces a final state which serves as an input to the decoder, which predicts the traffic.

To forecast multiple future timesteps, the paper leveraged the sequence-to-sequence architecture, which uses a encoder for embedding the whole input sequence into a final state, and uses this as an initial input for the decoder which predicts multiple steps ahead. The encoder and decoder are built with the proposed DCGRU units.

Datasets

The paper conducted experiments on the METR-LA dataset, a real-world large-scale traffic dataset. It contains traffic information from 207 sensors from the highways of Los Angeles County over 4 months (2012/03 ~ 2012/06).

We also evaluated the paper on another dataset, EXPY-TKY. This dataset contains traffic speed information from 2841 sensors on the highways of Tokyo for over three months (2021/10 ~ 2021/12).

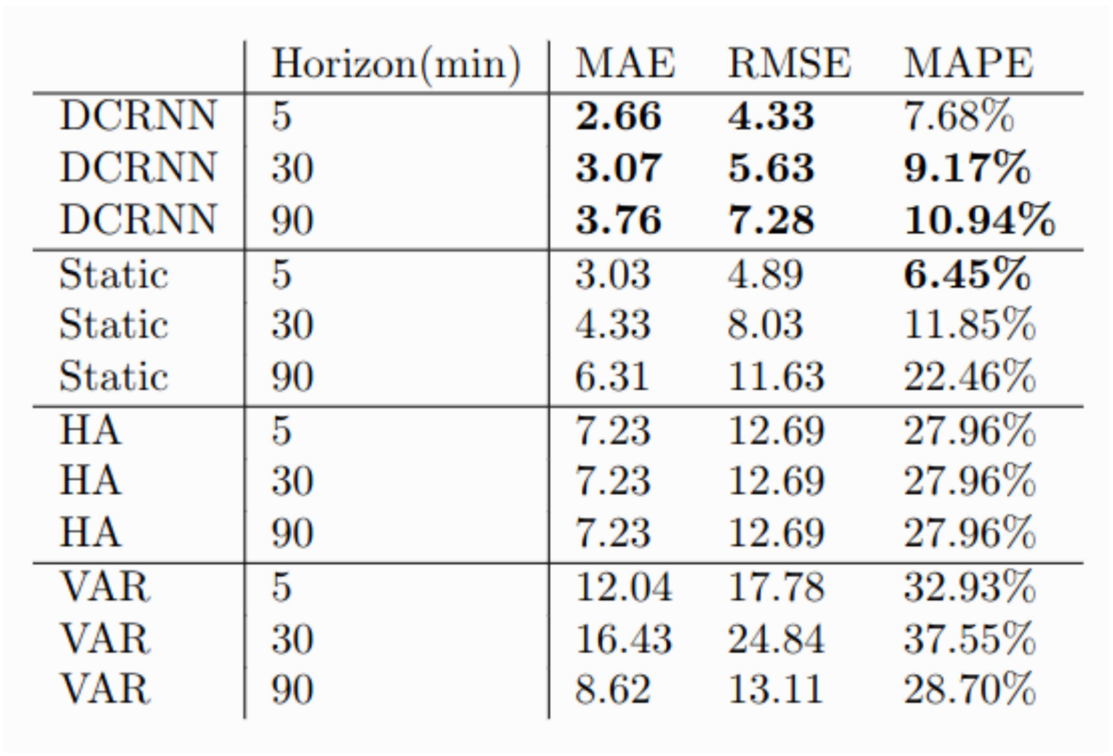
For both dataset, we aggregated traffic speed data into 5 minute windows, and applied z-score normalization. Moreover, we constructed the sensor graph which is a weighted directed graph containing spatial information of the road network. To construct the graph, we first computed the matrix which contains the distance between every pair of sensors. Then, the adjacency graph of the sensor is calculated by thresholding using a Gaussian kernel. Additionally, we set entries which are lower than a threshold to zero for efficient calculation.

Evaluation

This section covers the experimental results of our reproduction. The paper compares the proposed DCRNN architecture with several simple and state-of-the-art baseline models, for example: Historical Average (HA), Vector Auto-regressive model (VAR), Linear Support Vector Regression (SVR), Feed forward neural network (FNN) and Auto-Regressive Integrated Moving Average (ARIMA). However, the authors only provide implementations for HA and VAR as well as a static method, which simply outputs a prediction equal to the value seen a certain amount of timesteps ago. Implementing competing algorithms other than the DCRNN remains out of scope for this project, therefore we will use these three methods to compare the reproduced results.

To get valuable results during the reproduction we tested the proposed model as well as the baseline methods on two different datasets, namely the METR-LA dataset and the EXPY-TKY dataset. Due to computational constraints we only used a subset of the original dataset, more specifically data from 40 sensors and 4096 five-minute time steps compared to the original 207 sensors and 34271 timesteps on METR-LA.

When testing the EXPY-TKY dataset we also used 40 sensors and 4096 time steps.



	Horizon(min)	MAE	RMSE	MAPE
DCRNN	5	**2.66**	**4.33**	7.68%
DCRNN	30	**3.07**	**5.63**	**9.17%**
DCRNN	90	**3.76**	**7.28**	**10.94%**
Static	5	3.03	4.89	**6.45%**
Static	30	4.33	8.03	11.85%
Static	90	6.31	11.63	22.46%
HA	5	7.23	12.69	27.96%
HA	30	7.23	12.69	27.96%
HA	90	7.23	12.69	27.96%
VAR	5	12.04	17.78	32.93%
VAR	30	16.43	24.84	37.55%
VAR	90	8.62	13.11	28.70%

Table 1. - Performance comparison of different approaches for traffic speed forecasting. Bold values highlight the best performance. Dataset: METR-LA.

Table 1 shows the comparison of the 3 different baseline methods and the proposed DCRNN model. Looking at the values, it can be observed that the DCRNN method outperforms all other methods across all three loss functions for all three forecast horizons (5, 30, and 90), except for the MAPE where the static method performs better than the rest. This suggests that the DCRNN method is the most accurate and reliable forecasting method among the baselines. The most noticeable improvement is over the Vector Auto-regressive model, where we saw an improvement of over 300%. Overall we observe a 13% to 300% increase in performance of the DCRNN compared to the baselines.

Out of all the baseline methods the values indicate that the static model is the best in predicting future traffic speeds for all time horizons. Whereas the worst performer is the VAR.

	MAE	RMSE	MAPE
Paper	3.15	6.45	**8.8%**
Reproduced	**3.07**	**5.63**	9.17%

Table 2. - Comparison of Performance statistics between the results from the authors and the reproduced results. (30 minute Horizon)

The values of Table 2 show the MAE, RMSE and MAPE for both the reproduced and the results from the original paper. Both performance statistics were computed with a time horizon of 30 minutes. When comparing these values it seems that they are both in line with each other, reinforcing the validity of this reproduction in spite of the differences in implementation. The difference between the two may arise from the fact that we only reproduced the results on a smaller subset of the data.

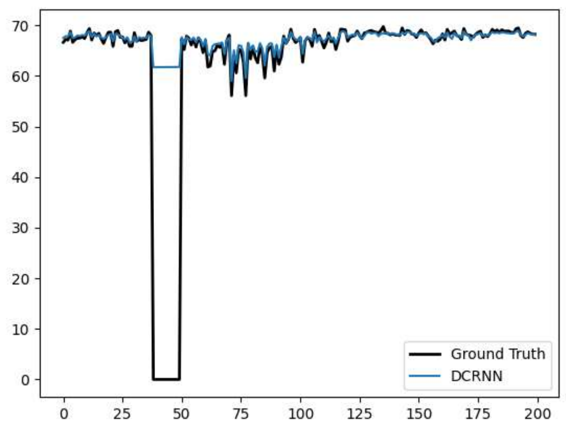


Figure 2a: DCRNN model, example traffic forecast for one sensor.

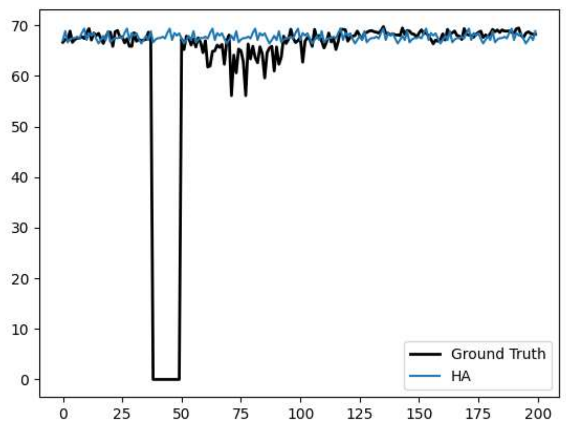


Figure 2b: Historical Average, example traffic forecast for one sensor.

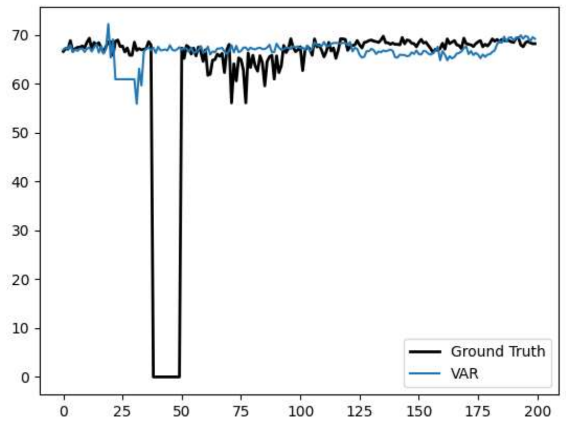


Figure 2c: Vector auto-regressive model, example traffic forecast for one sensor.

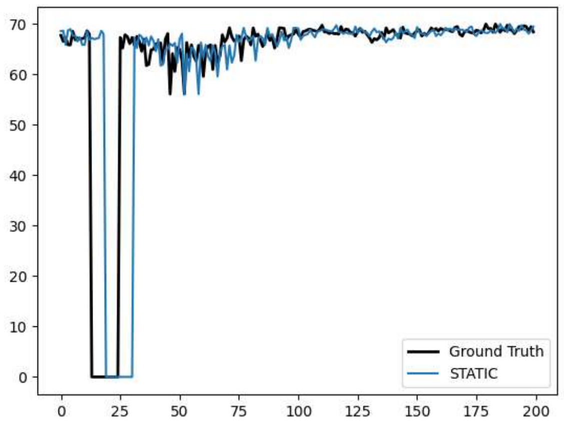


Figure 2d: Static method, example traffic forecast for one sensor.

Figure 2 depicts visualizations of the traffic speed forecasts for the baseline methods as well as the DCRNN model. The static method, as seen in Figure 2d, is the simplest method of the three baselines; it predicts the last observed value with a certain time lag. This way, it manages to capture the drop in traffic speed to zero quite well with a certain delay.

The Vector auto-regressive model on the other hand, as well as the historical average model, fails to capture the drop in traffic speed and generally handles the peaks and valleys pretty poorly.

Looking at the final model, the DCRNN in Figure 1a, it manages to capture the trend reasonably well. It only seems to struggle with peaks and valleys, where it underestimates both of them, this becomes very apparent when looking at the traffic speed drop to zero.

	Horizon(min)	MAE	RMSE	MAPE
DCRNN	5	**2.66**	**4.33**	**7.68%**
DCRNN	30	**3.07**	**5.63**	**9.17%**
DCRNN	90	**3.76**	7.28	**10.94%**
Static	5	4.30	6.55	14.98%
Static	30	4.57	6.64	16.18%
Static	90	5.25	7.45	18.98%
HA	5	3.86	5.76	16.25%
HA	30	3.86	5.76	16.25%
HA	90	3.86	5.76	16.25%
VAR	5	4.09	6.00	15.45%
VAR	30	3.80	5.69	15.74%
VAR	90	3.86	**5.68**	15.93%

Table 3. - Performance comparison of different approaches for traffic speed forecasting. Bold values highlight the best performance. Dataset: EXPY-TKY.

Table 3 shows the comparison of the 3 different baseline methods and the proposed DCRNN model on the EXPY-TKY dataset. Looking at the values, it can be observed that again the DCRNN method outperforms all other methods across all three loss functions for all three forecast horizons (5, 30, and 90), except for the RMSE this time where the Vector auto-regressive model performs better than the rest with regards to a forecast horizon of 90 minutes. This suggests that the DCRNN method is the most accurate and reliable forecasting method among the tested methods.

Surprisingly, the baseline methods in general perform a lot better on the EXPY-TKY dataset compared to METR-LA despite using a smaller subset of data for both of them.

We observe on average a 30-50 percent increase in performance for the DCRNN model compared to the baselines.

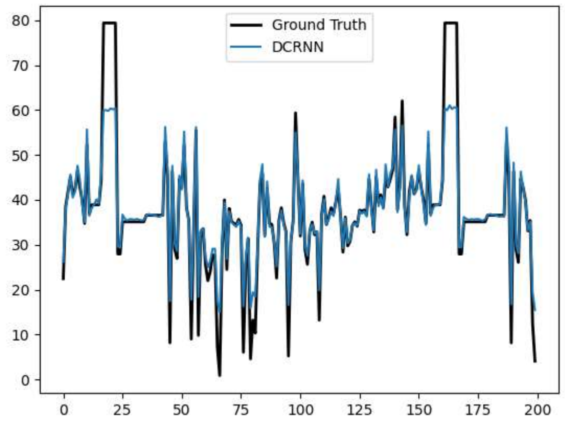


Figure 3a: DCRNN model

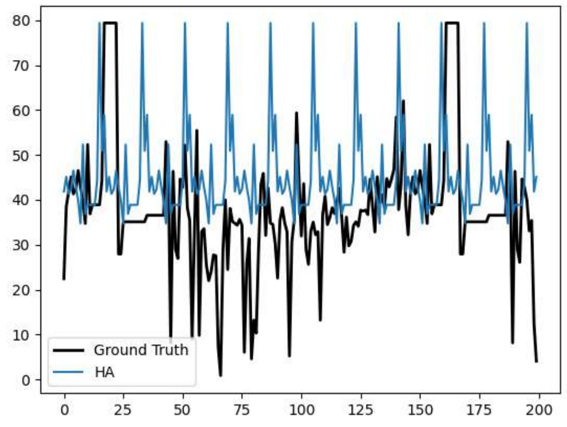


Figure 3b: Historical Average

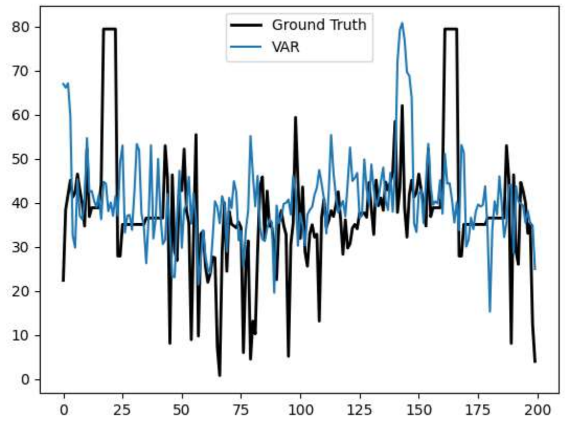


Figure 3c: Vector Auto-regressive model

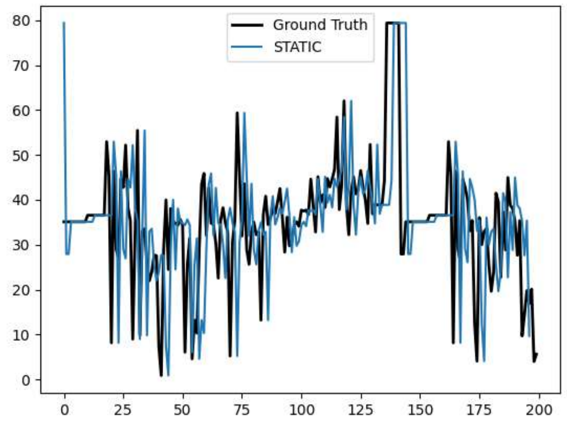


Figure 3d: Static Method

Figure 3 shows four visualizations of traffic speed forecasts on the EXPY-TKY dataset. Once again, we can see that the DCRNN give the best predictions of future traffic speeds, but undershoots the peaks and valleys.

Conclusion

In this project it was confirmed that the DCRNN architecture performed better overall than the baseline methods for which the authors provided implementations. A significant improvement has been demonstrated on the original dataset and a new dataset of traffic information, further reinforcing the authors claims. Exact percentage increases in performance could not be perfectly replicated, both from a lack of access to competing algorithms and possibly to the limitations introduced by the computational restrictions. These factors could be addressed by further research in order to arrive at more reliable quantitative results.

References

[1] Li, Yaguang, et al. "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting." arXiv preprint arXiv:1707.01926 (2017).

[2] https://paperswithcode.com/dataset/expy-tky

[3] https://paperswithcode.com/dataset/metr-la