FORECASTING TRAFFIC FLOW [2]

Jacob Stollznow (s182071), Jan Hurt (s181123), Luka Kovac (s182214), Guo Ziyu (s181706)

DTU Compute
Technical University of Denmark

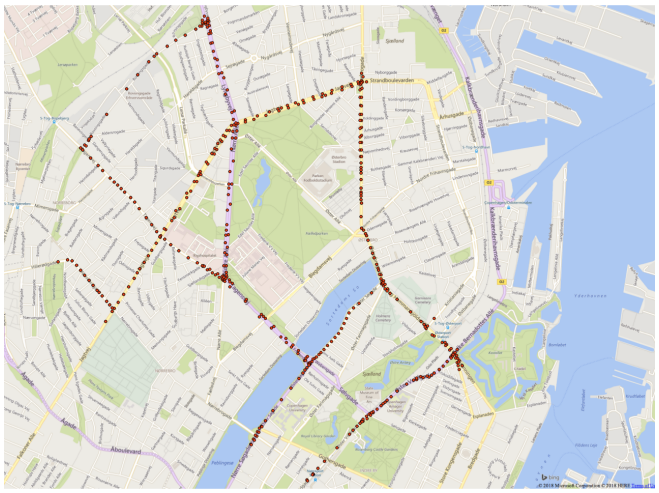


Fig. 1. The location of the measurement points on a map of Copenhagen.

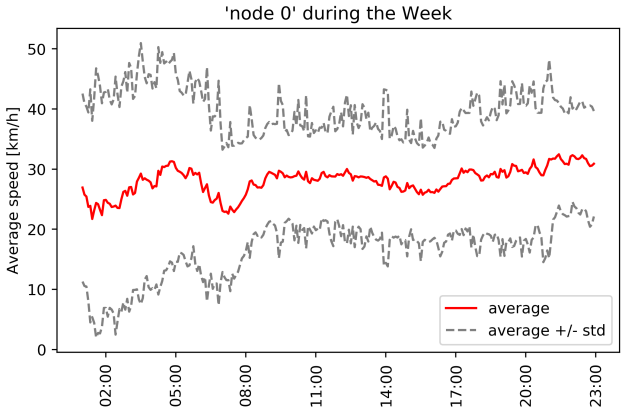


Fig. 2. Comparison between the average speed measurement and the standard deviation during the week.

1. INTRODUCTION

Data provided by Google was used to forecast the average speed of traffic in Copenhagen. Various neural networks were implemented and were than compared to baselines and the DCRNN implementation on Github [5]. The performance of each network was evaluated at a various number of prediction steps, the number of future steps determined how far into the future the network was predicting. Using these performance results the comparable performance of each network was justified. Finally, a conclusion was drawn reflecting the best performing network for traffic predictions at each given varying future steps.

2. DATA DESCRIPTION

Each node in the data set provided consisted of two values taken at five minute intervals each day. Every five minute the two values collected were average speed and flow bucket value. The measurements by Google included data from each 'dot'(node/road segment) on the map presented in figure 1, there were 270 nodes in total.

The average speed was self-explanatory and specified the average speed of the vehicles which had traveled past the given node in the past five minute interval. The flow bucket value was relative and was a value comparable between nodes in the system. Flow bucket values ranged from zero to ten, with nine representing a node with traffic flow in the top 10% of all nodes. Whilst zero represents a node with traffic flow in the bottom 10% of the nodes. [4] Since during the day-hours the flow bucket was almost always at 9, this was not used as a prediction input or output.

3. DATA CONFIGURATION

3.1. Data Cleaning

Initial observations of the data provided by Google shows significant noise in the data at various nodes on many different days. This noise in the data was evident in the average speed variability illustrated in figure 2 for node 0. Through observation of the figure it was evident the data required appropriate cleaning to yield good network performance. In which case it was decided the data was to be limited to daylight hours, namely between 5:00 and 21:00. In which case this selection of data included the morning traffic rush (approximately 5:30

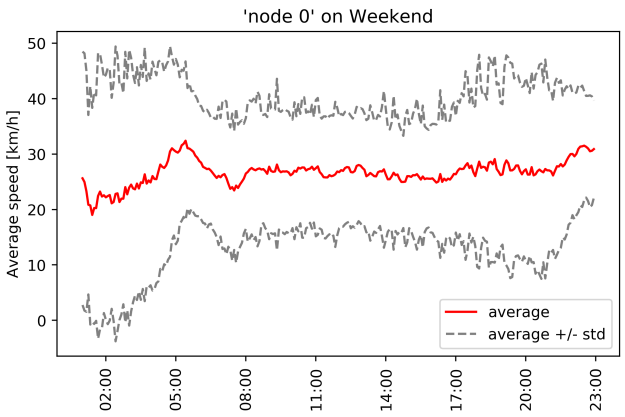


Fig. 3. Comparison between the average speed measurement and the standard deviation during the weekend.

to 8:30) and the afternoon traffic rush (approximately 13:00 to 17:30). In limiting the data set to the specified daylight hours, it eliminated some of the seemingly 'glitch' values in which the average speed for intervals overnight was occasionally $162kph$.

The next reduction involved the removal of weekends, this was because weekends did not adhere to the same patterns which were present during the week and thus were eliminated to simplify the prediction of traffic flow. This difference in traffic flow was supported by figure 3 illustrating the traffic flow on a weekend. Through comparison of figure 2 and figure 3, the given days did not follow the same traffic flow and could limit the performance of the implemented neural networks.

Some days within the data had missing values for some time intervals, these intervals were given values through linear interpolation. This was applied to the average mean speed and seemed like a logical solution considering the manner in which the data was constructed and measured.

3.2. Data Organization

Following the various data cleaning steps, the data was reconfigured for easy manipulation within each network. Using the reduced data set, steps were carried out to aid the implementation of normalization, batching, training, and consequent performance.

The data was organized to reflect the way in which the performance of each network would be evaluated. The performance of each network was to be evaluated according to their predictions at some selected prediction intervals. In which case the data set was re-organized such that each given interval in each day of each node included two arrays. One array included the 12 previous interval values (i.e. the traffic intervals for the past hour) and the other array reflected the 12 future intervals (the traffic intervals for the next hour). Where the first future interval reflected the true value of the interval it represented. Using this data set it was simplistic to alter the prediction steps and thus carry out the required loss measurements to evaluate performance.

4. NETWORK IMPLEMENTATION

Several neural networks were implemented to predict traffic flow in Copenhagen. Each network tested varied in either hyper-parameters or architecture altogether. Most neural networks implemented were optimized for best performance through techniques provided throughout the year.

To ensure the performance of each network was comparable the data preparation implemented by each network was maintained. This ensured all alterations in performance reflected the designed neural network.
The networks implemented are listed below:

- LSTM network
- GRU network
- 1 layer fully connected network considering 1 node
- 1 layer fully connected network considering 1 node with historical averages as separate input
- Simple fully connected network considering multiple nodes
- DCRNN

The first two networks, the LSTM and GRU neural networks were designed in a very similar fashion. The LSTM network was far more complex to train and optimize often taking more than double the time to train or test network hyper-parameters. In addition changes in the hyper-parameters and other potential improvements to the network architecture did not see an increase in performance. Due to these implications the LSTM network was difficult to optimize.

Conversely, the GRU network was simplistic to train requiring far less computational power than the similar LSTM network. The GRU network saw improvement in performance as the number of hidden units was altered and the fully connected layer parameters were varied.
Both networks did not see improvement through the use of dropout layers, regularization or additional fully connected layers.

The simple fully connected networks differed in style and architecture. The first listed considered the data from only 1 node. This was very easy to optimize and was used as a

baseline, which the other networks were compared to.

However, this network already performed reasonably well, in which case, another 1 layer network was developed which used the historical averages of the training data-set as an input.

The last network listed is the DCRNN. This paper [5] provided some code on github [1], which was essentially used as a black-box and trained on AWS. This network used the relationship between the different measurement-points on the streets and applied this information to the training of the network. The relationship of the points on the network was fed in the network in the form of the distances on street. To get those distances, the relationship between each point had been noted per hand. Then the shortest path between each point on the street was calculated using the networkx library [3].

The network was then trained on AWS using the p2.xlarge instance. Since a large amount of nodes was used for training (270), the training took a significant amount of time (approx. 7 days for 83 epochs).

5. NETWORK PERFORMANCE

To evaluate and compare the performance of each designed network the mean squared loss error was compared at various prediction intervals. A range of prediction intervals were investigated including 1,3,7 and 12, where one interval is a five minute period. The table below presents the results obtained when the selected prediction intervals were investigated. The results can be found in the table 1.

Network	1	3	7	12
FC 1 layer	10.59	24.53	31.16	34.65
Historical average	28.63	28.41	28.21	28.75
FC 1 node average	11.59	21.28	23.39	25.23
FC 2 nodes average	11.19	22.12	24.59	26.25
FC 3 nodes average	12.38	22.58	25.36	27.19
LSTM	14.89	24.95	29.11	33.33
GRU	37.89	40.74	42.48	45.70
DCRNN	8.54	20.84	24.57	27.92

Table 1. Performance of the networks ($MSE[(km/h)^2]$) for different prediction horizons ($\cdot 5min$).

For certain networks, the performance was also visualized in a graph. This graph compares the MSE value with the prediction step, this was important to visualize how each network changes depending on how far into the future it was required to predict the flow of traffic. This can be found in figure 4.

5.1. Discussion of the performance of the networks

As can be seen in figure 4 the implemented neural networks, which are not using the historical averages as a separate input,

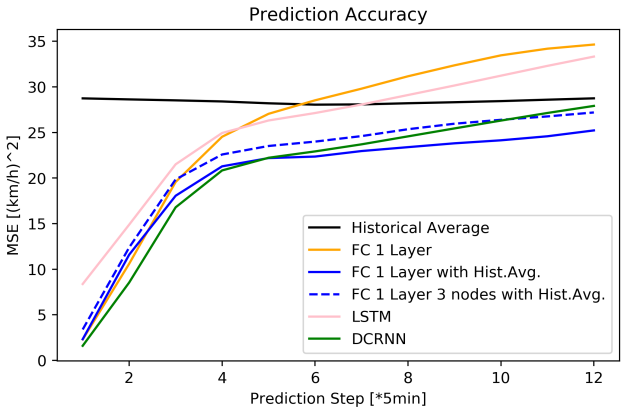


Fig. 4. Accuracy of some networks, depending on the prediction horizon.

are suffering from a performance drop-off after around 25-30min. i.e.: their performance gets worse than the historical average.

To prevent trend, the fully connected 1 layer network was re-implemented using historical averages. For a prediction horizon >20min, even a very simple 1 layer 'network' (if this can be even called a neural network, or rather a multivariate moving average model) outperforms the very complex DCRNN, which took quite some time to train. This was not further investigated nor applied the other networks due to the computational requirements to train the various networks but could be further investigated to observe its influence on performance.

However, the DCRNN outperforms all networks for the smaller prediction horizons as supported by the results. This suggests the DCRNN network has a deeper understanding of the street network, as it can extract more information from the data.

A visualization of these predictions for 1 day inside the testing set can be found in figure 5. Using this graph, the difference between the networks can be analyzed even further. One observable difference, the DCRNN shows far more 'confidence' in its predictions than some of the other networks. The other, more simple networks are much more conservative with their predictions, and especially for a longer prediction horizons, they tend to underestimate the magnitude of the traffic congestion (how low the average speed is). Observing the predictions by the implemented networks, they do not seem as sensitive as the DCRNN network to changes in traffic conditions. This was not a desirable effect as the networks were still seemingly effected even though they were not appropriately sensitive to changes in traffic conditions.

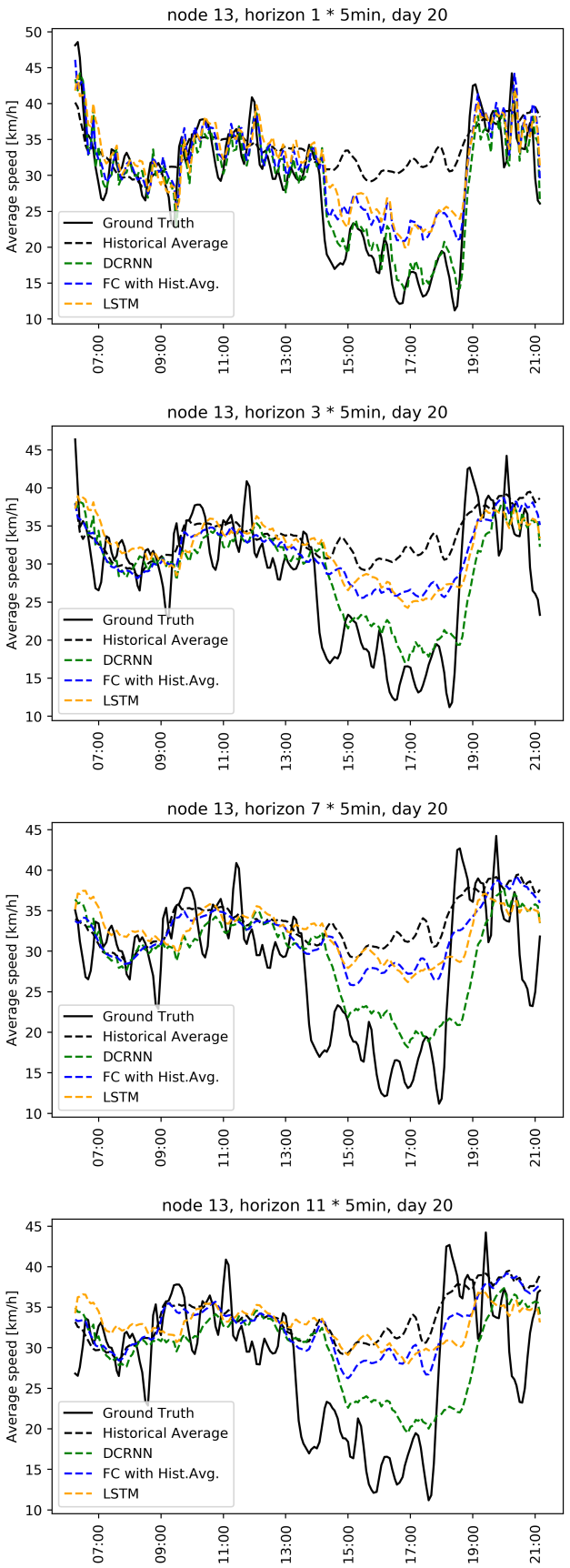


Fig. 5. Predictions for different prediction horizons for 1 day inside the testing data-set.

6. DISCUSSION

Traffic prediction could improve the flow of society and have a significant social and economic impact on people's lives. Research of this topic can greatly impact every city and has the potential to achieve tasks which currently seem impossible, such as the synchronization of all traffic lights in populated cities. Currently, in most of the cities traffic lights are without sensors or any prediction and thus create unnecessary traffic congestion.

For our use, we got traffic data around Nørre Campus during a whole day. Since the data was noisy, the data was filtered and cleaned assuming this would lead to more accurate traffic predictions. This approach involved the selection of specific times and days of the week for the training data-set. Selecting a different method of data cleaning would have altered the prediction accuracy, whether it be good or bad.

Due to complexity and time constraints, only the average speed was used in all neural networks. As previously mentioned traffic density of a given segment or intersection was measured in term of a given 'flow_bucket' value. Using both speed and flow could increase the performance of traffic forecasting. For increased accuracy of predictions, other variables could be configured into the network to deepen the relationship between nodes and relations to training data, such as the seasonal trends of traffic (tourist season, holidays, etc.).

Considering the results presented, it can be seen how many different networks were trained to predict traffic flow through the average speed prediction. Only simpler models were used for training and the best results varied as the prediction interval varied.

Other papers approach this problem with different types and higher complexity neural networks such as spatiotemporal graph convolutional nets and diffusion convolutional recurrent neural nets. For now, these networks provide one of the best results in traffic forecasting by using spatial and temporal dependencies of nodes.

Starting with simpler recurrent neural networks and conducting research offers a good possibility of discovering new and better ways to traffic forecasting.

A limitation in the design of each neural network exists in the sense that none of the designed neural networks accounted for the nodes relative positions. Thus, when training every node at the same time via one LSTM, it was slow and unstable, because this approach neglects connections between nodes. Each node is directly influenced by nodes around it, so we must consider nodes next to the target node, and also nodes around these nodes because they are affected too. Because of that, we can introduce Graph LSTM to improve predicting accuracy. Graph LSTM unfolds the graph to a tree with the target node at root, and establish one LSTM for nodes at each depth. A top-level LSTM summarizes output of these LSTMs and produces the prediction of the target node. Graph LSTMs emphasize the effect of distance and links between nodes,

which is the key to traffic prediction. The further a node is from the target node, the less its effect on the target node, and such a relationship is perfectly accounted for in Graph LSTM.

7. CONCLUSION

Predictions were made for traffic data provided by Google for Copenhagen based on the average speed at each five minute interval of the day. Through this process, it was realized how much easier was to fit networks for just one node. To be able to get more realistic and applicable results, the simple fully connected layers neural network was developed to a network which connected multiple nodes. Feed-forward network did not record good performance, but other complex networks such as the LSTM network did.

Attempts with different, more complex networks which trained on more data and involved 'flow bucket' values could have provided better prediction interval performance and should be further investigated.

8. REFERENCES

[1] Github: liyaguang/dcrnn. https://github.com/liyaguang/DCRNN.

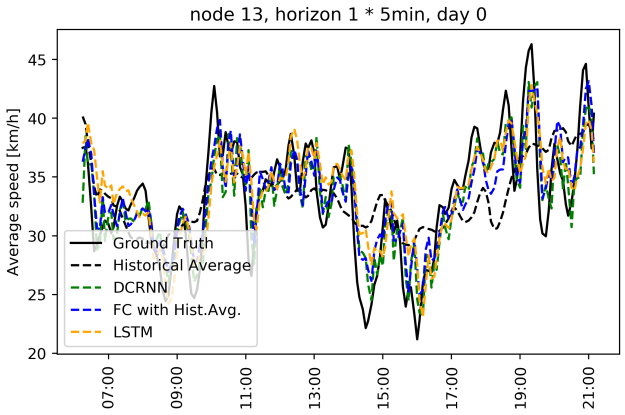
[2] Github repository of the project. https://github.com/iqopi/dtu-2456-Traffic_Forecasting_Copenhagen.

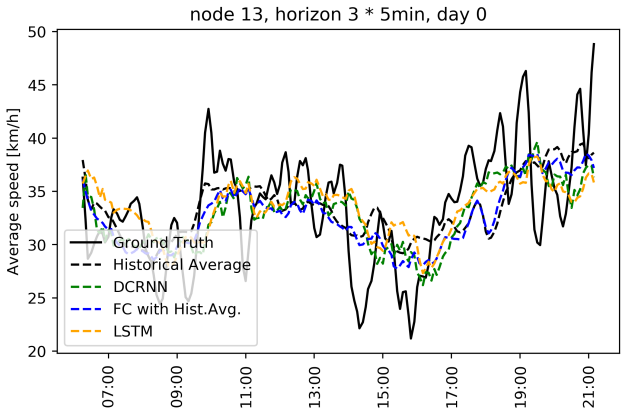
[3] Networkx. https://networkx.github.io/.

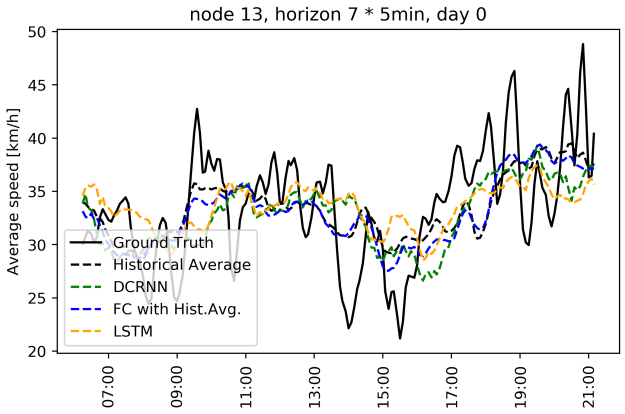
[4] Google. Getting started guide data description, release process, tools, and examples, 2016.

[5] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Graph convolutional recurrent neural network: Data-driven traffic forecasting. *CoRR*, abs/1707.01926, 2017.

9. APPENDIX


node 13, horizon 1 * 5min, day 0
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM


node 13, horizon 3 * 5min, day 0
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM


node 13, horizon 7 * 5min, day 0
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM

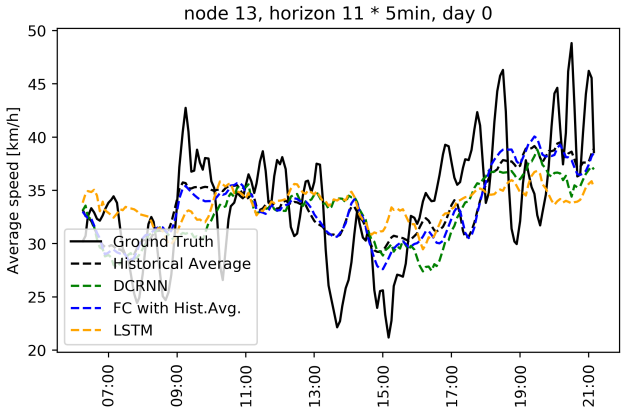
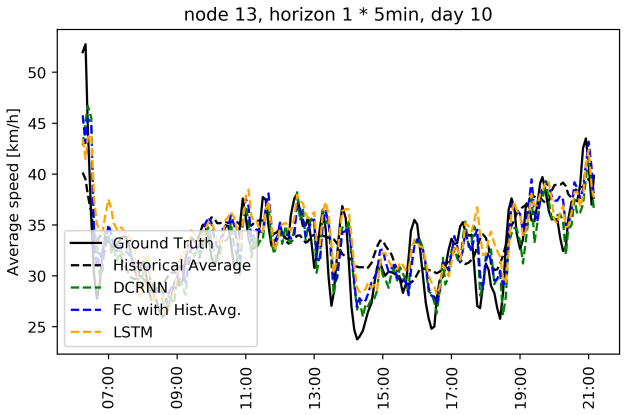
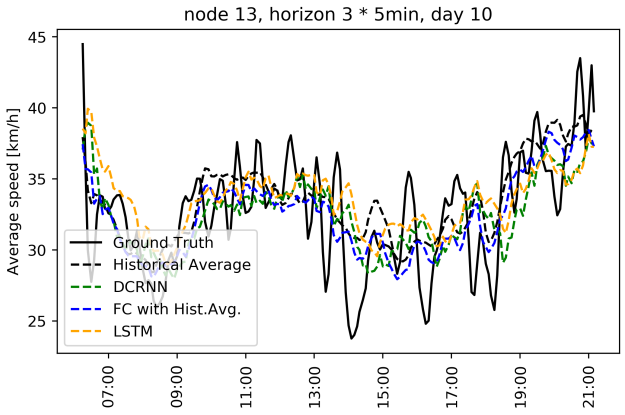
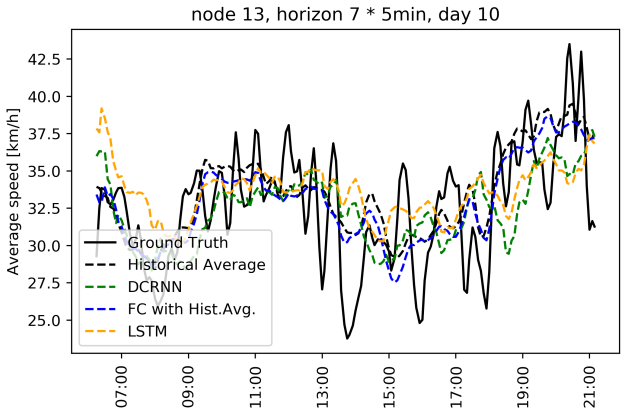

node 13, horizon 11 * 5min, day 0
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM

Fig. 6.


node 13, horizon 1 * 5min, day 10
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM


node 13, horizon 3 * 5min, day 10
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM


node 13, horizon 7 * 5min, day 10
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM

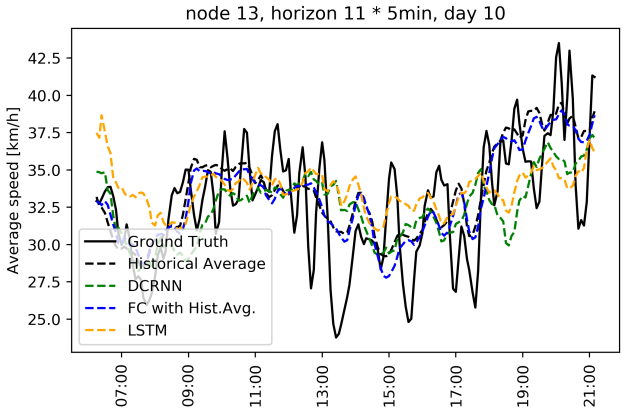

node 13, horizon 11 * 5min, day 10
Average speed [km/h]
Ground Truth
Historical Average
DCRNN
FC with Hist.Avg.
LSTM

Fig. 7.