STTF Project
Final Presentation

Member:

Bilal Muhammad

Nicolas Knauber

Hwapyeong Yu

TA:

Yuyol Shin

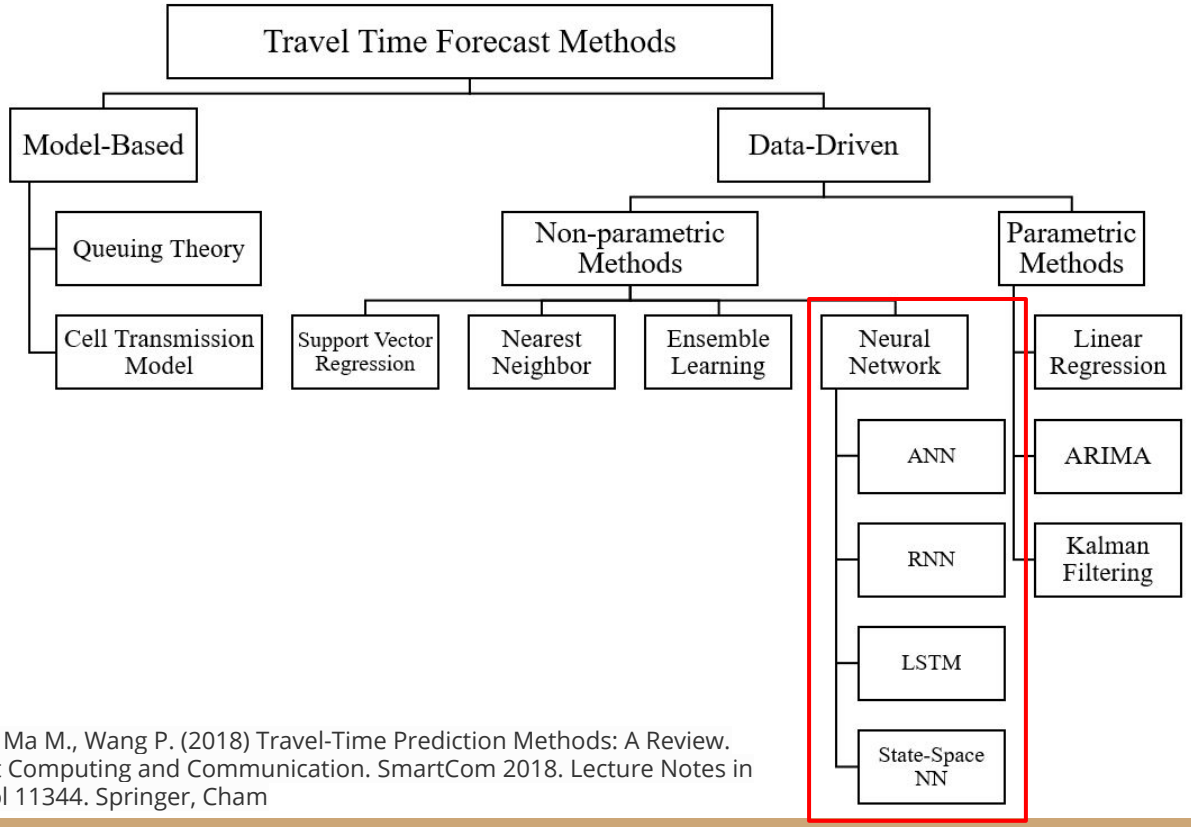
Prof.:

Yoonjin Yoon

Contents

Introduction

Short Term Traffic Forecasting



Source: Bai M., Lin Y., Ma M., Wang P. (2018) Travel-Time Prediction Methods: A Review. In: Qiu M. (eds) Smart Computing and Communication. SmartCom 2018. Lecture Notes in Computer Science, vol 11344. Springer, Cham

Advantages of Deep Learning in STTF

- ✓ *Hou et al. (2018)* said deep learning model can make accurate predictions for all the segments in the transportation network with a single model structure, instead of building customized models for each segment separately.

- ✓ Various traffic phenomena that are difficult to explain using model-based methods can be included in the deep learning prediction results.

- ✓ Data process and computing techniques have been developing which can make it available to learn large amounts of traffic data collected on a city scale.

Source: Hou, Yi, and Praveen Edara. "Network scale travel time prediction using deep learning." *Transportation Research Record* 2672.45 (2018): 115-123.

Research Object

- ✓ Implement deep learning methods on the travel time forecast and improve the accuracy of the prediction.

- ✓ Compare the performance of deep learning/traditional methods and draw the pros and cons of each methods.

Literature Review

Traditional Methods for STTF

1. ARIMA
   - ✓ *Alghamdi et al. (2019)* used ARIMA model for analyzing non-stationary and non-normally distributed traffic data and make prediction of traffic congestion. Many preprocessing steps were performed before the application of ARIMA model.
2. KNN
   - ✓ *Bustilos et al. (2011)* proposed travel time prediction model includes single-NN for the freeflow state and multiple-NN for the congested state. This hybrid-NN model predicted the highway travel time with real-time speed.

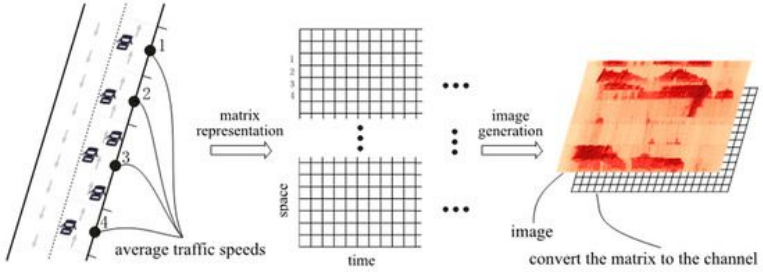
Alghamdi, Taghreed, et al. "Forecasting Traffic Congestion Using ARIMA Modeling." *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*. IEEE, 2019.
Bustillos, Brenda I., and Yi-Chang Chiu. "Real-time freeway-experienced travel time prediction using N-curve and k nearest neighbor methods." *Transportation Research Record* 2243.1 (2011): 127-137.

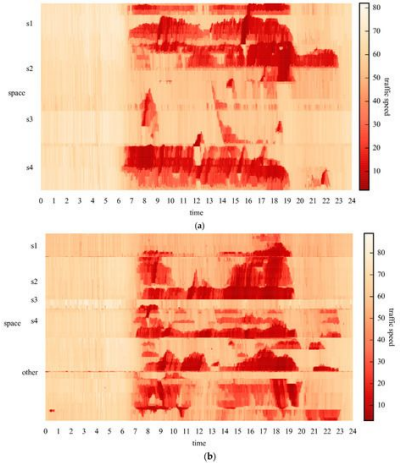
Deep Learning for STTF

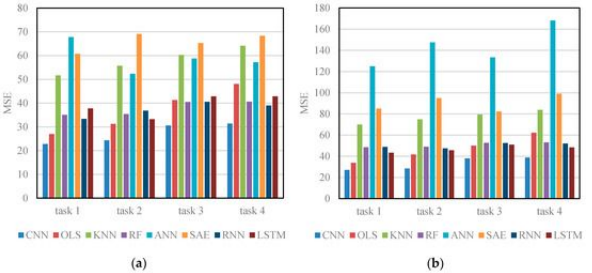
1. SVM
   - ✓ *Yao et al. (2017)* proposed a short-term traffic speed prediction model which is based on the SVM. The SVM exhibited good performance compared with an ANN, KNN, a historical data-based model, and a moving average data-based model.
2. CNN
   - ✓ *Ma et al. (2017)* used CNN to learn traffic as images and predicts large-scale, network-wide traffic speed with a high accuracy. Spatiotemporal traffic dynamics were converted to images describing the time and space relations of traffic flow via a two-dimensional time-space matrix.
3. LSTM
   - ✓ *Ma et al. (2015)* presented a novel long short-term memory neural network to predict travel speed using microwave detector data. LSTM outperformed other algorithm in terms of accuracy and stability. Travel speed prediction performance improves as the time lag becomes longer.

Yao, Baozhen, et al. "Short-term traffic speed prediction for an urban corridor." *Computer-Aided Civil and Infrastructure Engineering* 32.2 (2017): 154-169.
Ma, Xiaolei, et al. "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction." *Sensors* 17.4 (2017): 818.
Ma, Xiaolei, et al. "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data." *Transportation Research Part C: Emerging Technologies* 54 (2015): 187-197.

Learning traffic as **images**: a deep convolutional neural network for large-scale transportation network speed prediction







Limitations and research goal

Limitations of previous work:

- ✓ They focussed on one type of the road, but actual urban travel is usually distributed on various type of the road (highways, arterials and bridges etc.).
- ✓ Comparison measurement was only RMSE or MAPE value which cannot find out which model has less accuracy in what traffic conditions and roads.
- ✓ The effects of the ordering of links and the speed limit on the prediction accuracy were not considered.

Research Goal

- ✓ Speed prediction on mixed geometry with highways and urban roads
- ✓ Various attempts to increase prediction accuracy including link ordering and speed limit
- ✓ Pros and cons of each prediction methods based on the traffic conditions and roads

Methodology

Data Description

Data Type	5 min average speed by link
Collected Area	Seoul, the Republic of Korea
Collected Date	01/04/2018~30/04/2018
Link Property	Road type Start/End point Speed limit

YEAR	MONTH	DAY	HOUR	MIN	LINKID	SPEED	ROADNAME	STARTPOINT	ENDPOINT	DIRECTION	LENGTH	ROADTYPE	SPEEDLIMIT
2018	4	1	0	0	1.2E+09	19.78	올림픽대로	영동대교	청담대교	Up	586	Primary	60

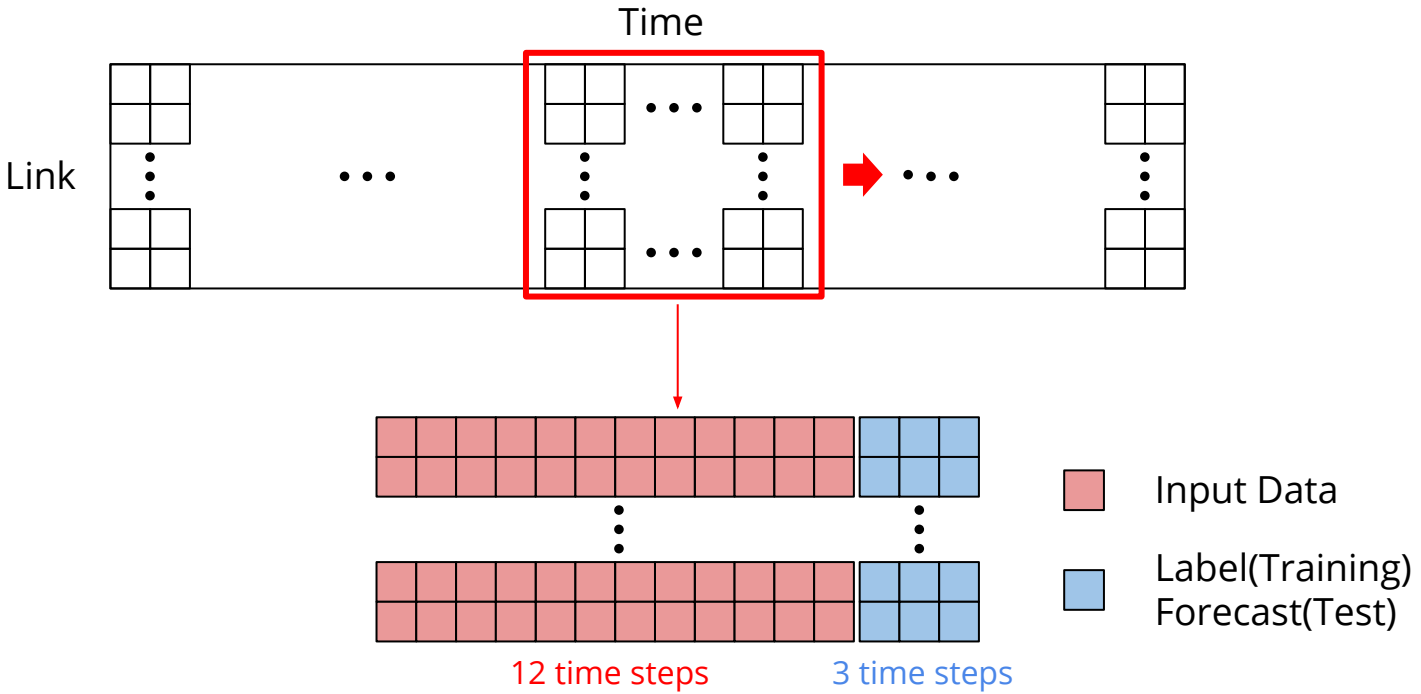
Study Site

Study Site 1
Highway, 10 links

Study Site 3
Urban, 44 links

Study Site 4
Highway+Bridge+Urban,
26 links

Speed Forecast Task

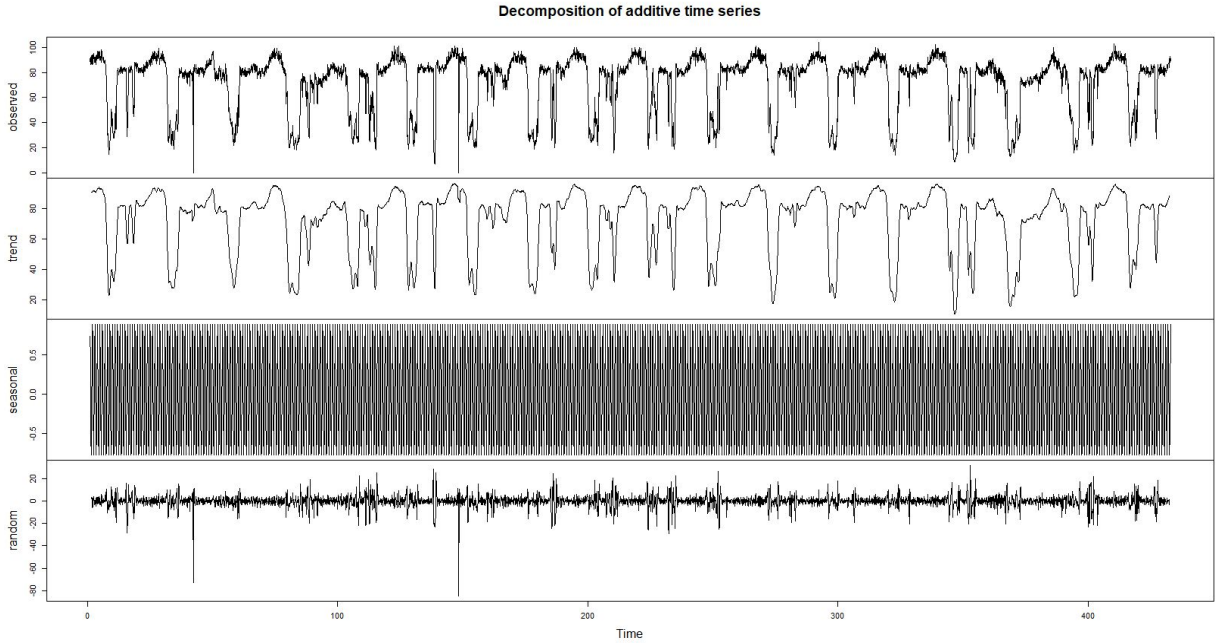


Forecast Methods- Traditional Statistical Models

- ✓ **Mean method** (mean of past values)
- ✓ **Naïve method** (last observed value).
- ✓ **Seasonal naïve method** (Forecasts equal to last value from same season).
- ✓ **Drift method** (last value plus average change).
- ✓ **Exponential smoothing model** (uses an exponentially decreasing weight for past observations).
- ✓ **ARIMA model** (Auto Regressive Integrated Moving Average)

Data Set Construction for traditional models

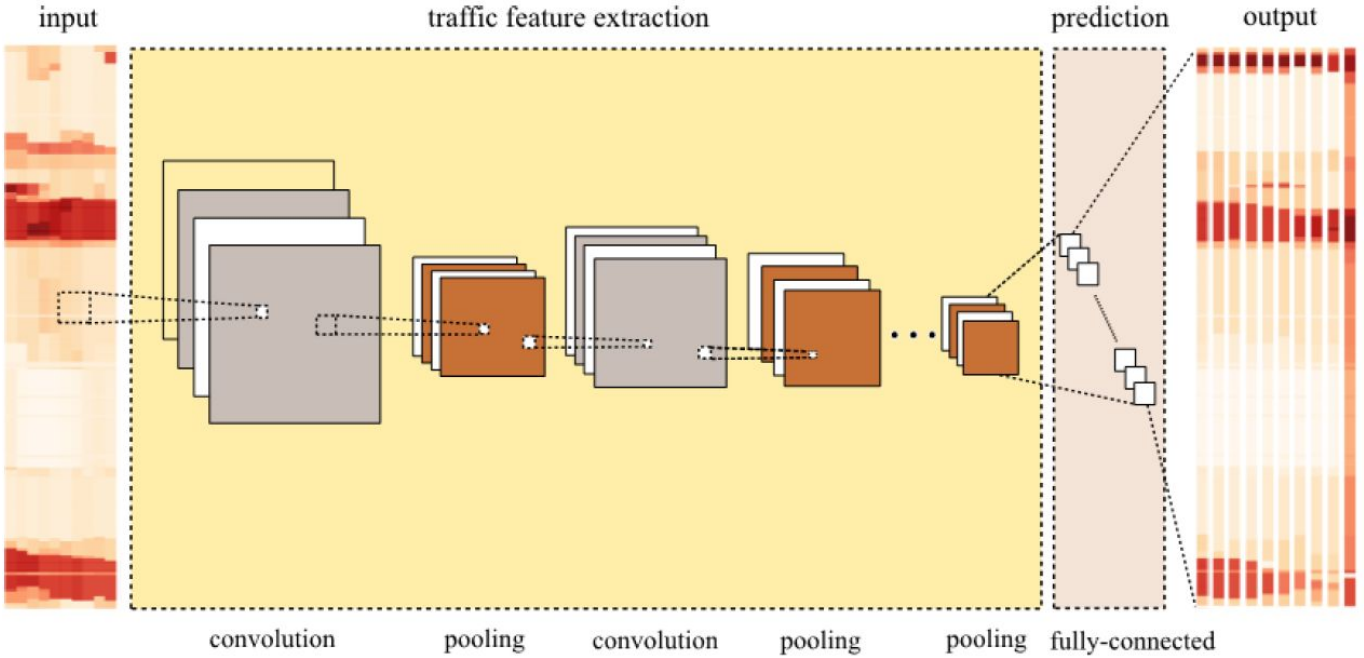
- ✓ Seasonality value = 12 (1 hr data, 12's 5 min slots)
- ✓ Training data = 5184 slots of 5 min data (18 days x288 slots per day) = 432 hrs
- ✓ Test data = 3 days



Forecast Methods-KNN

1. Data Set Construction
   - ✓ Historical data set(10 Links * 15 time steps, 273/day *15 days = 4095 set)
   - ✓ Test data set(273/day *6 days = 1638 set)
2. Euclidean Distance Calculation
   - ✓ Select a test data set and calculate euclidean distance of 1 hour data(12 time steps) with every historical data
   - ✓ Select the k historical data set with the minimum distance
3. Speed Forecast
   - ✓ Calculate an inverse distance weighted average speed with the k-nearest historical data set

Forecast Methods-CNN



Source: Ma et al. (2017), *Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction*

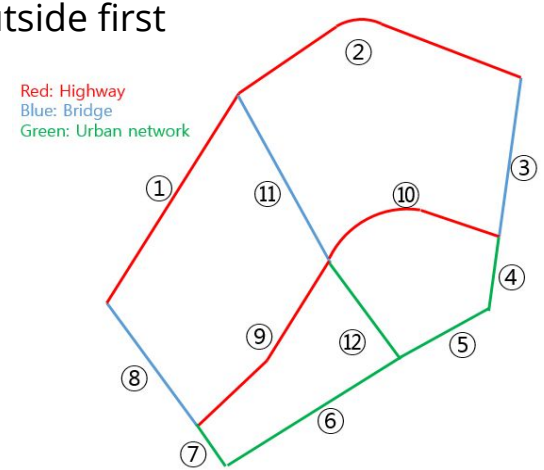
Forecast Methods-CNN

1. Hyperparameter
   - ✓ 2 convolution & pooling layer
   - ✓ 2 *v.s.* 3 dense layer
   - ✓ Dropout Layer(*p=0.2*) *v.s.* no dropout
   - ✓ Kernel Size = 3
2. Setting
   - ✓ ‘SAME’ padding
   - ✓ Activate Function = ‘ReLU’ for hidden layers and ‘linear’ for output activation
   - ✓ Loss Function = Mean Squared Logarithmic Error
   - ✓ Optimizer = Nadam

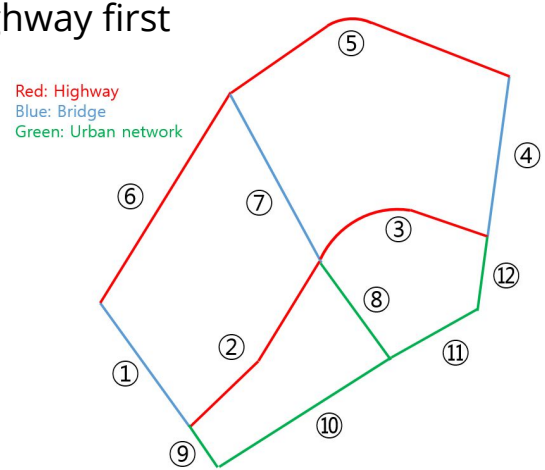
Forecast Methods-CNN

3. Data Representation
   - ✓ Only speed as input *v.s.* Add relative speed(speed/speed limit) as additional input
4. Link Ordering
   - ✓ Outside first *v.s.* Highway first *v.s.* Random ordering

Outside first

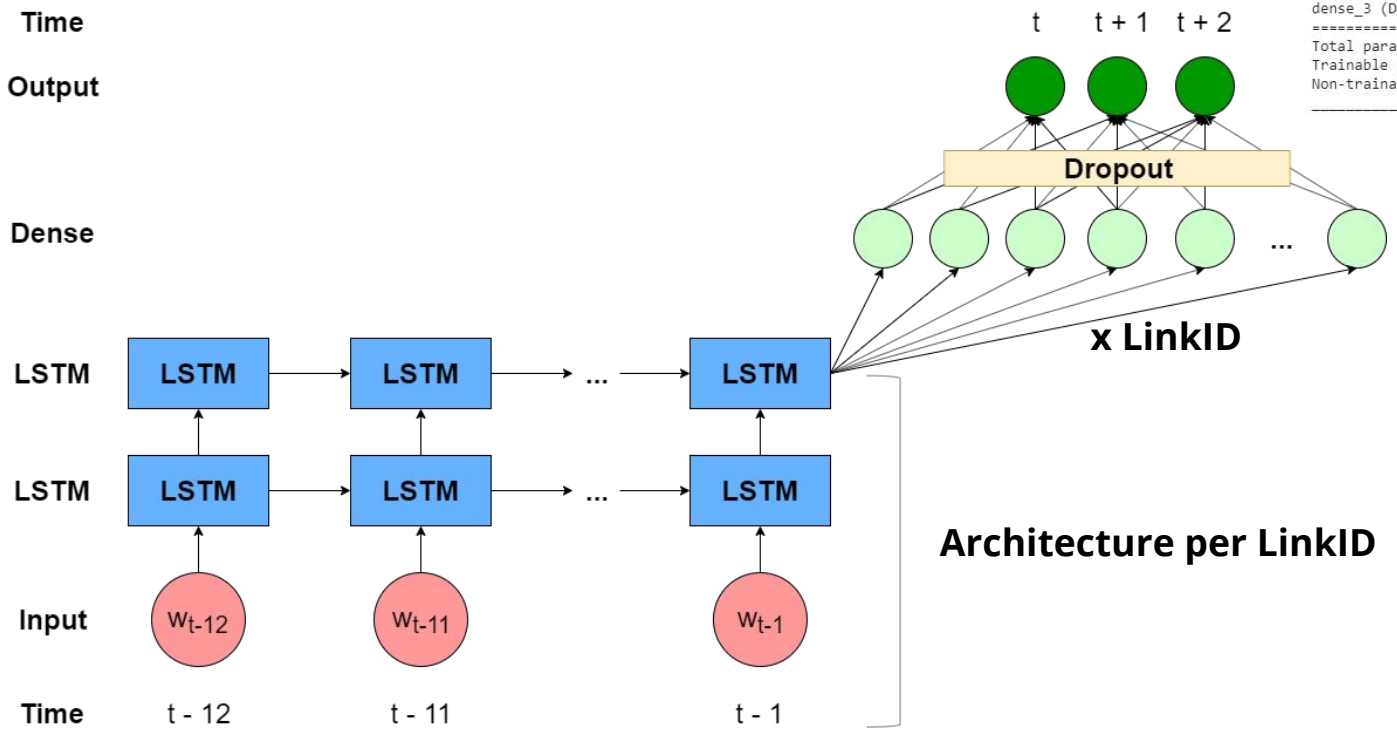


Highway first

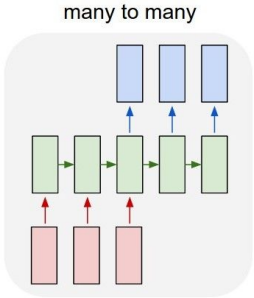


Forecast Methods-LSTM

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 12, 44)            15664
_________________________________________________________________
lstm_3 (LSTM)                (None, 44)                15664
_________________________________________________________________
dense_2 (Dense)              (None, 100)               4500
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_3 (Dense)              (None, 132)               13332
=================================================================
Total params: 49,160
Trainable params: 49,160
Non-trainable params: 0
_________________________________________________________________
```



Architecture per LinkID



Result

Result comparison of traditional models

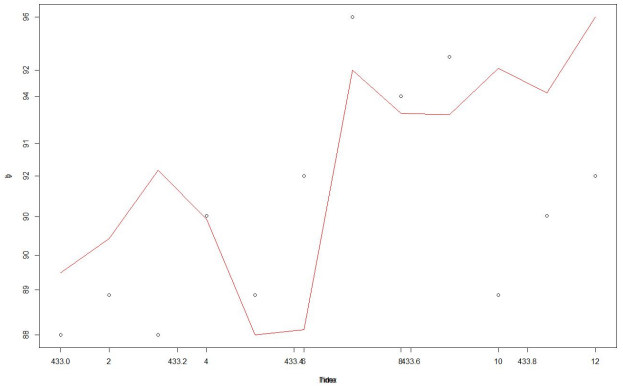
- ✓ ARIMA models are more general than exponential smoothing.
- ✓ ARIMA model fits the training data slightly better than the ETS model, but that the ETS model provides more accurate forecasts on the test set.

	Seasonal naive method	Exponential smoothing model	ARIMA model
Residuals	7.94	5.64	5.70

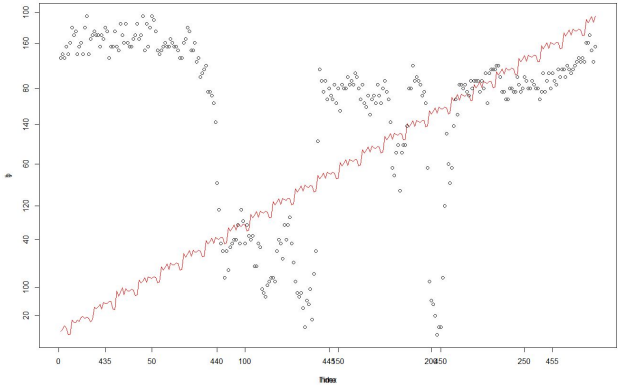
ARIMA Model result

Prediction Time	MAPE(%)
1 hour	2.2
2 hours	2.51
9 hours	39.3
1 day	111

1 hour forecasting

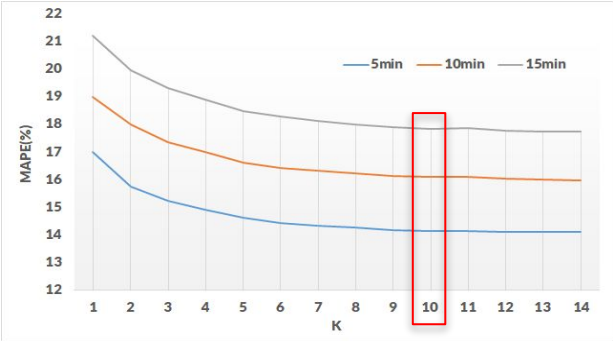


1 day forecasting

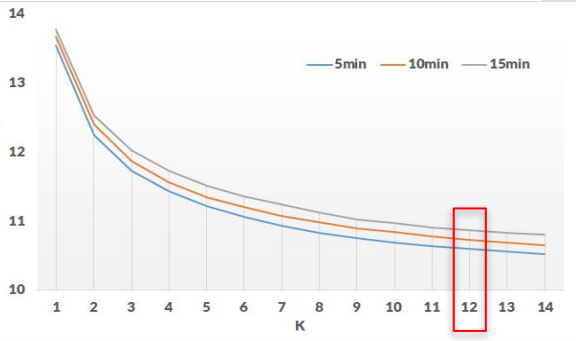


KNN- K Selection

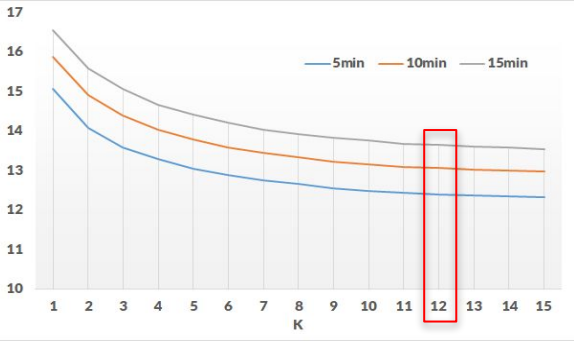
Study Site 1



Study Site 3

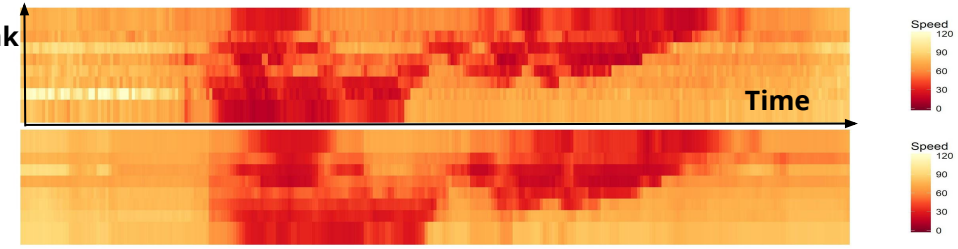


Study Site 4



KNN Result

Study Site 1

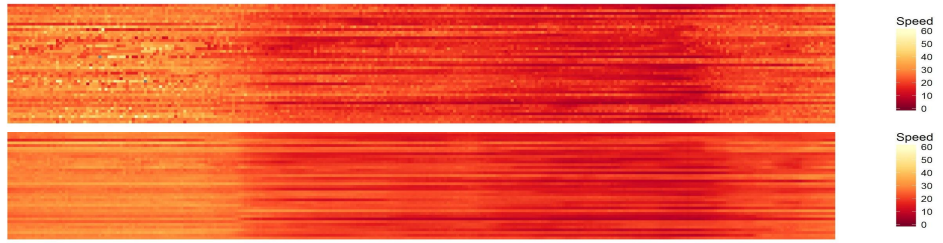


True

Pred

MAPE = 14.15%

Study Site 3

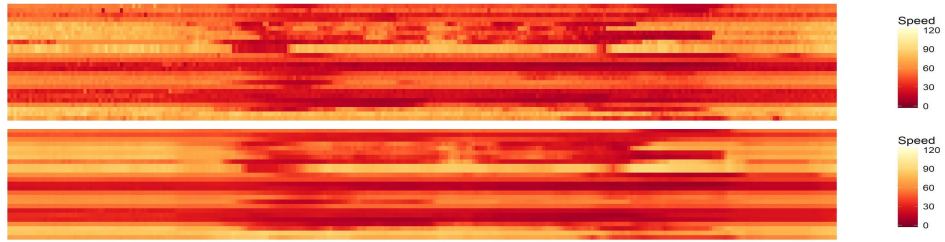


True

Pred

MAPE = 10.60%

Study Site 4



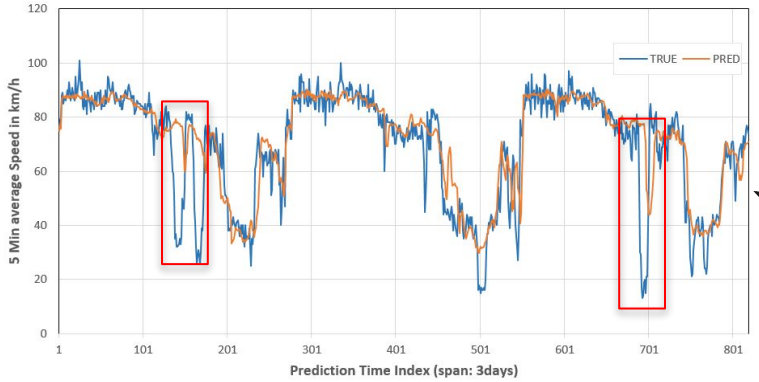
True

Pred

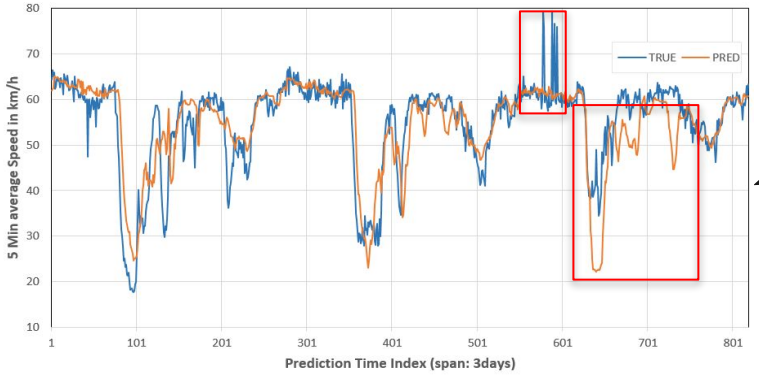
MAPE = 12.40%

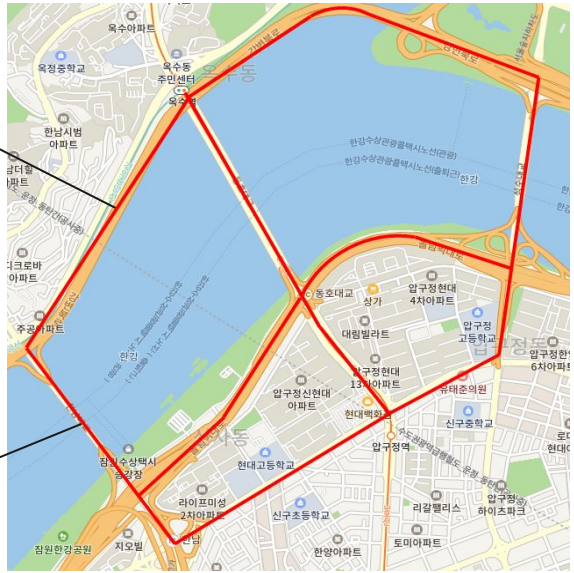
KNN Result - Highway & Bridge

강변북로



한남대교

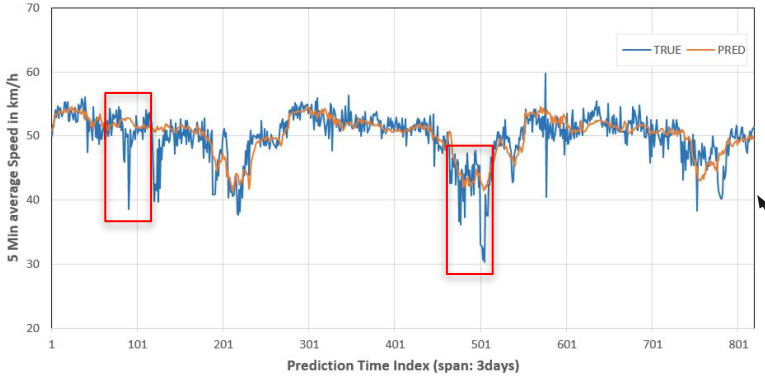




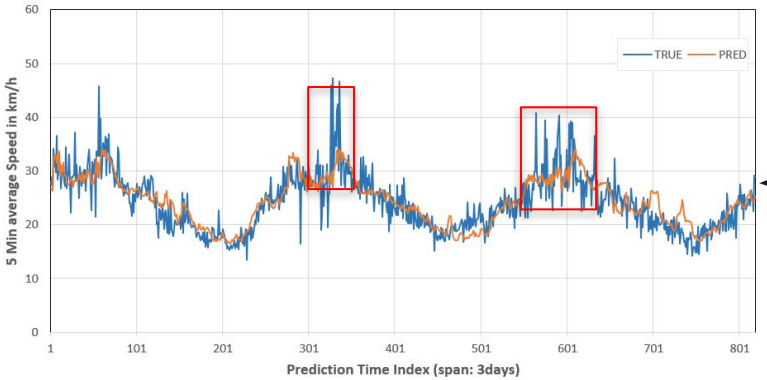
Study Site 4

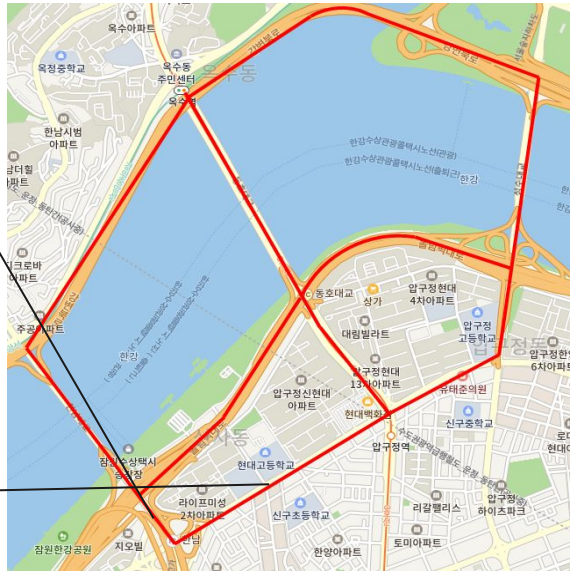
KNN Result - Urban

한남IC 부근



압구정로





Study Site 4

CNN Improvement-Summary

Improvement Method	Test Site
Add 1 Dropout Layer	Study Site 1
Remove 1 Dense Layer	Study Site 1
Train/Validation/Test Set Split	Study Site 1
Add Relative Speed as Additional Input	Study Site 4
Change Link Ordering	Study Site 3, 4

CNN Baseline Model

Ma et al. (2017)

Table 2. Hyperparameters of the CNN.

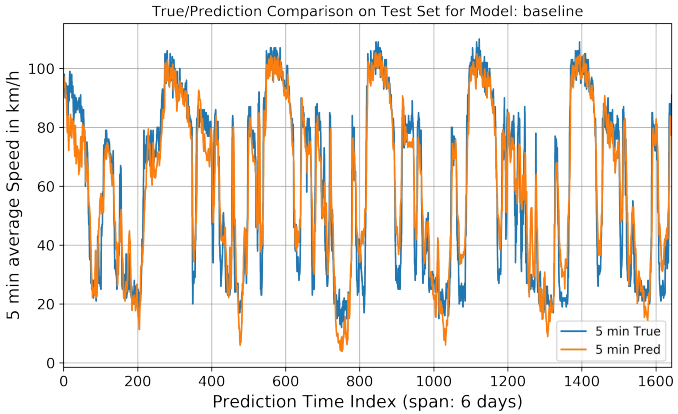
Layer	Name	Parameters	Dimensions	Parameter Scale
Input	—	—	(1, 236, 20)	—
Layer 1	Convolution	Filter (256, 3, 3)	(256, 236, 20)	2304
Layer 1	Pooling	Pooling (2, 2)	(256, 118, 10)	0
Layer 2	Convolution	Filter (128, 3, 3)	(128, 118, 10)	1152
Layer 2	Pooling	Pooling (2, 2)	(128, 59, 5)	0
Layer 3	Convolution	Filter (64, 3, 3)	(64, 59, 5)	576
Layer 3	Pooling	Pooling (2, 2)	(64, 30, 3)	0
Layer 4	Data flatten	—	(5760,)	0
Layer 4	Fully-connected	—	(1180,)	6,796,800
Output	—	—	(1180,)	—

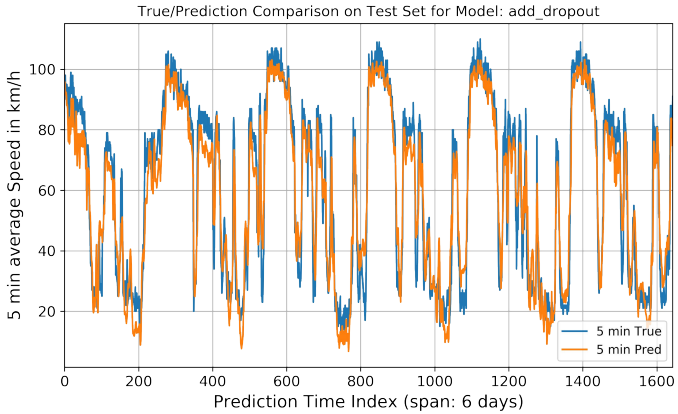
Baseline Model

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 10, 12, 256)       2560
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 5, 6, 256)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 5, 6, 128)         295040
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 2, 3, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 768)               0
_________________________________________________________________
dense (Dense)                (None, 128)               98432
_________________________________________________________________
dense_1 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_2 (Dense)              (None, 30)                1950
=================================================================
Total params: 406,238
Trainable params: 406,238
Non-trainable params: 0
_________________________________________________________________
```

CNN Improvement - Add Dropout Layer

- Dropout Value = 0.2
  - Aim: Preventing model from Overfitting
  - Functionality: Training - ignore nodes by *p*, Testing - use all Activations but multiply them with (1-*p*)





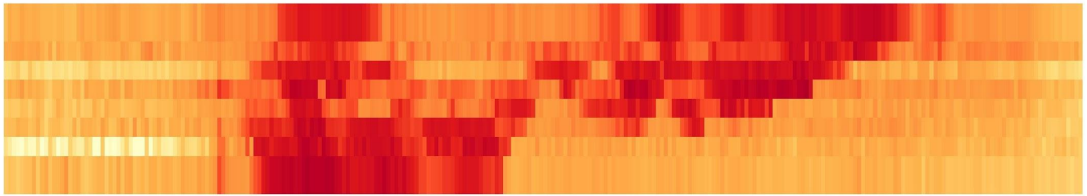
No Dropout, MAPE = 19.12%

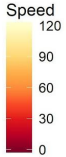
Add Dropout, MAPE = 16.98%

Study Site 1

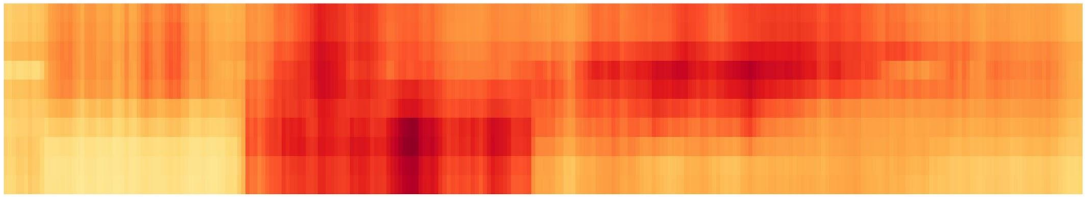
CNN Improvement - Remove 1 dense layer

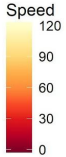
True data





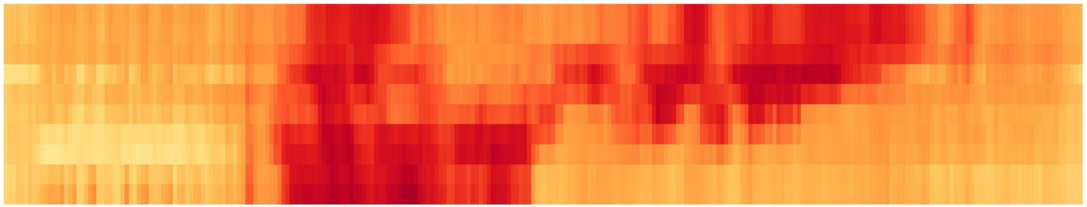
3 dense layer

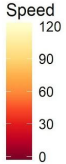




MAPE = 16.98%

2 dense layer

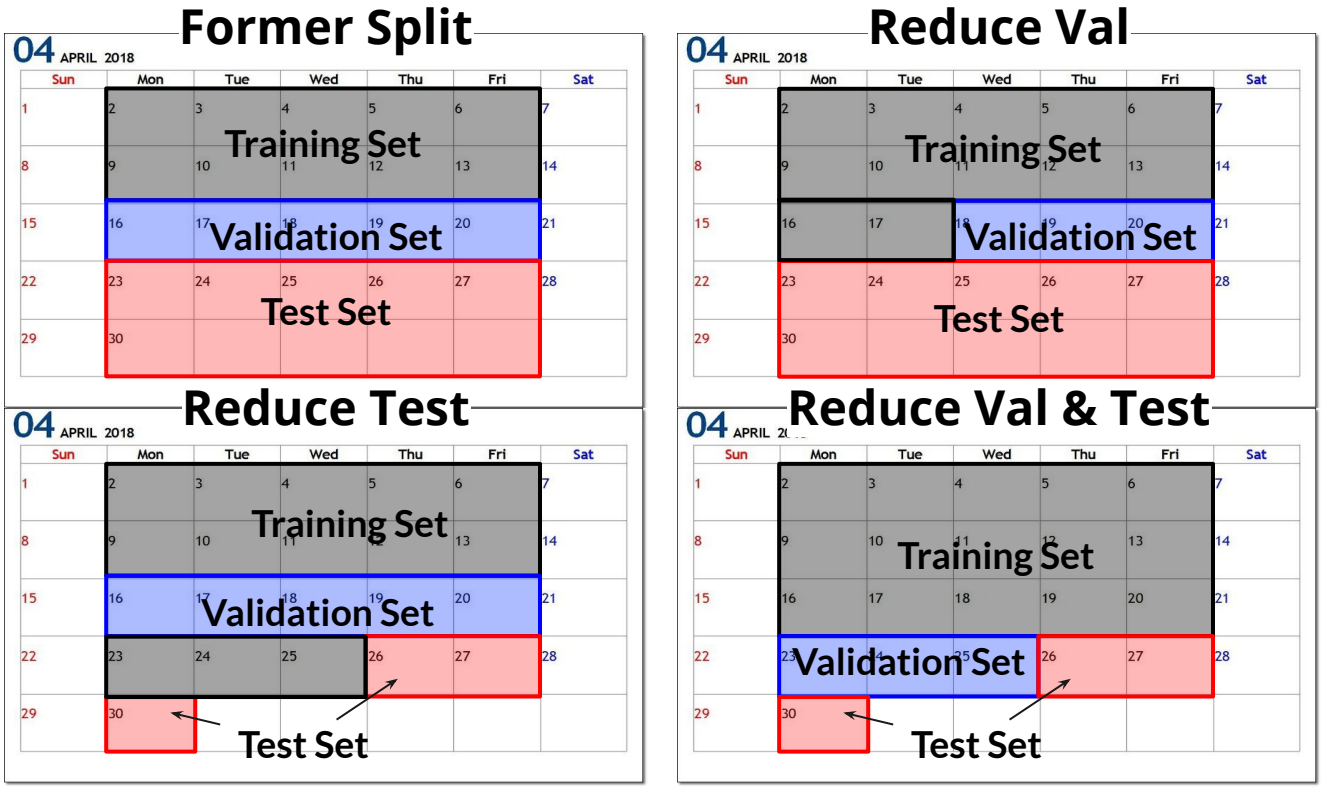




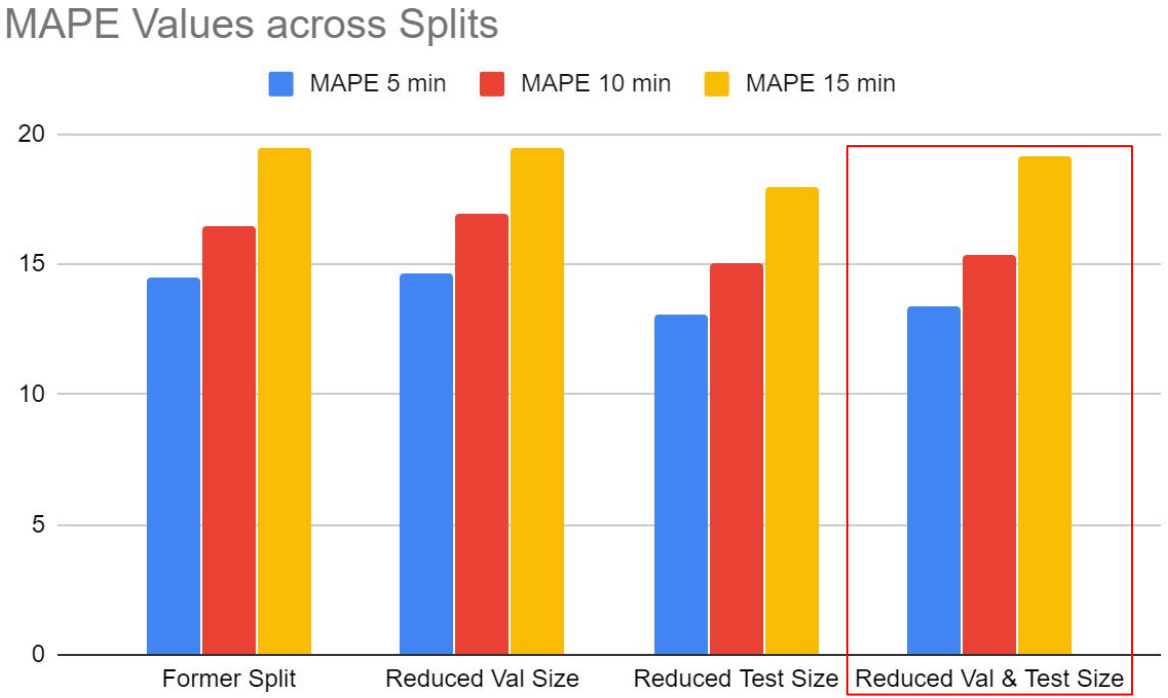
MAPE = 14.77%

Study Site 1

CNN Improvement- Train, Val., Test set Split



CNN Improvement- Train, Val., Test set Division

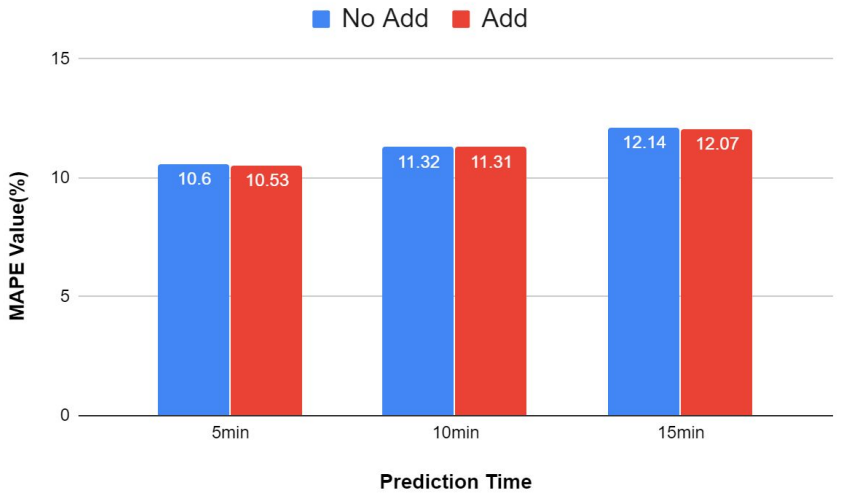


Study Site 1

CNN Improvement- Add relative speed as input

Idea: Adding the speed limit data could improve the performance

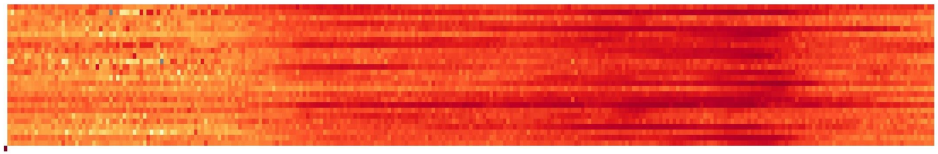
- ✓ Relative speed = average speed/speed limit
- ✓ 2D CNN with **additional channel** - experimented with & without 1×1 convolutional layer



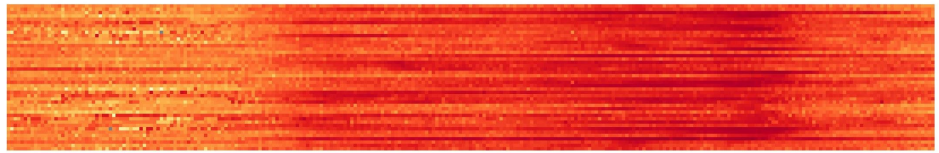
Study Site 4

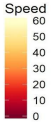
CNN Improvement- Link ordering(Study Site 3)

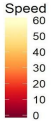
Random Order

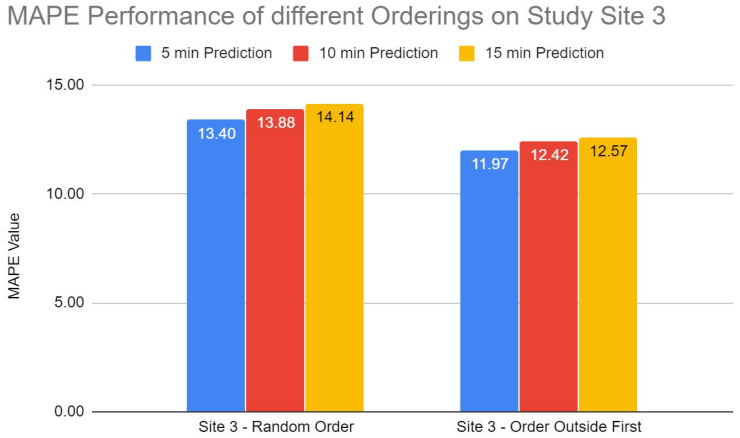


Outside First





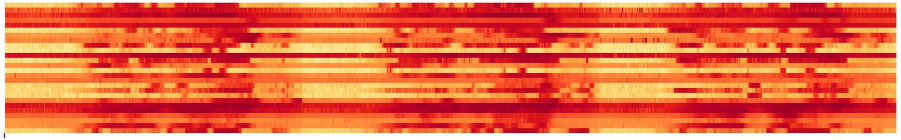


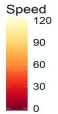


Study Site 3

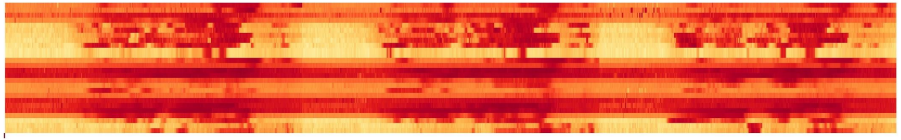
CNN Improvement- Link ordering(Study Site 4)

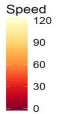
Random Order



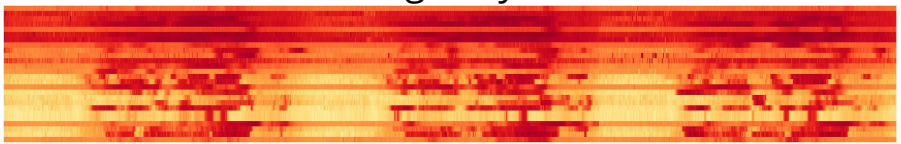


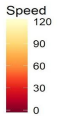
Outside First

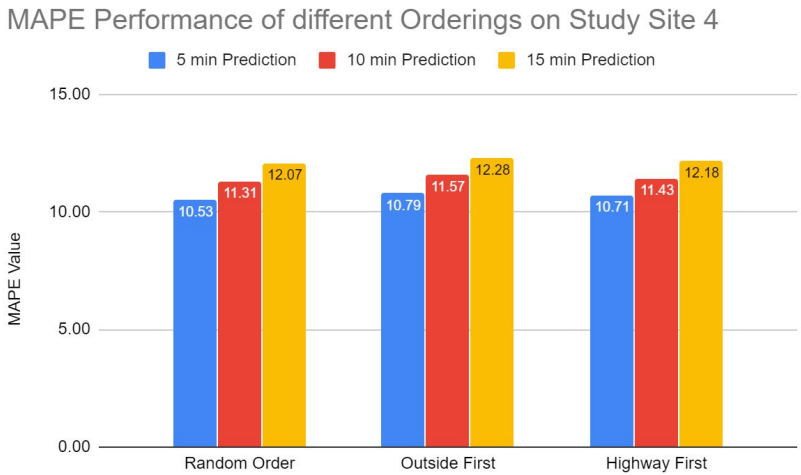




Highway First



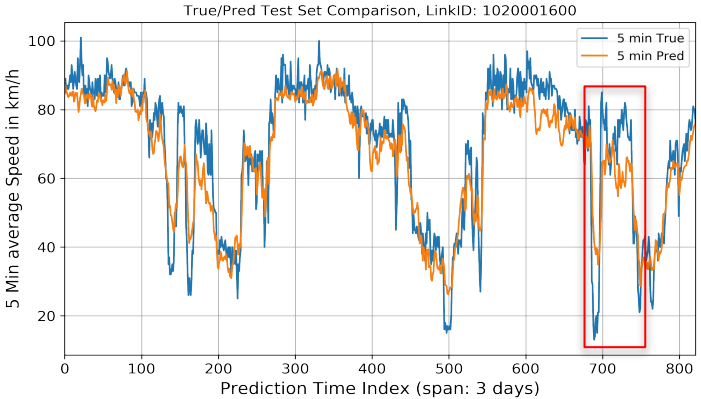




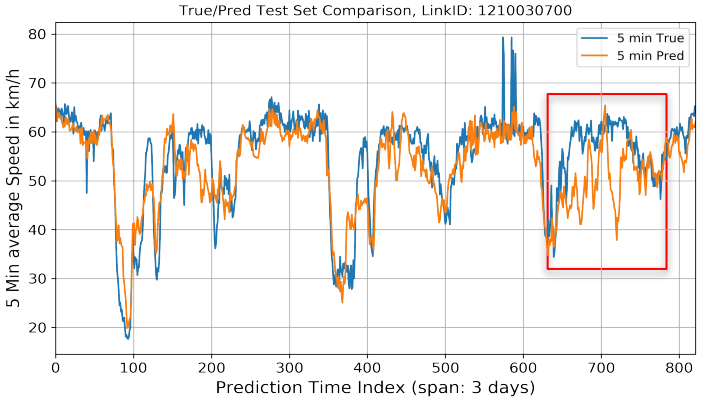
Study Site 4

CNN Result- Highway & Bridge

강변북로
Highway



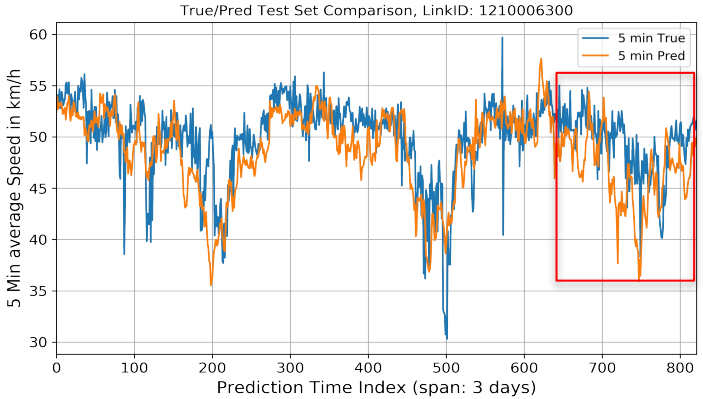
한남대교
Bridge



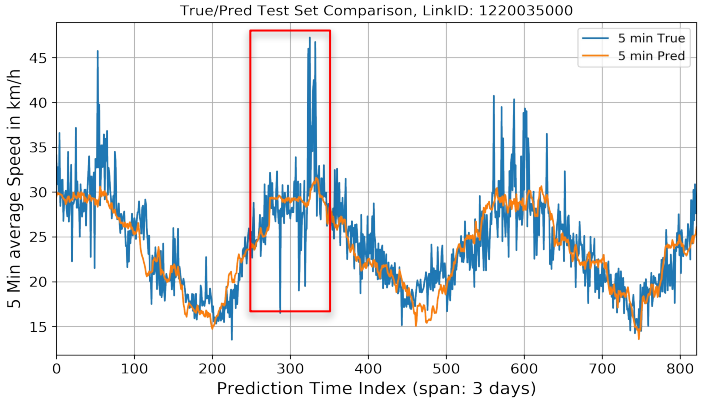
Study Site 4

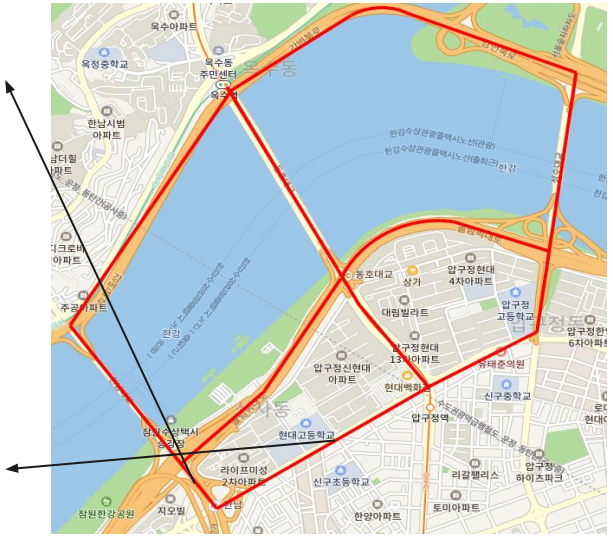
CNN Result- Urban

한남IC 부근
Near
Highway



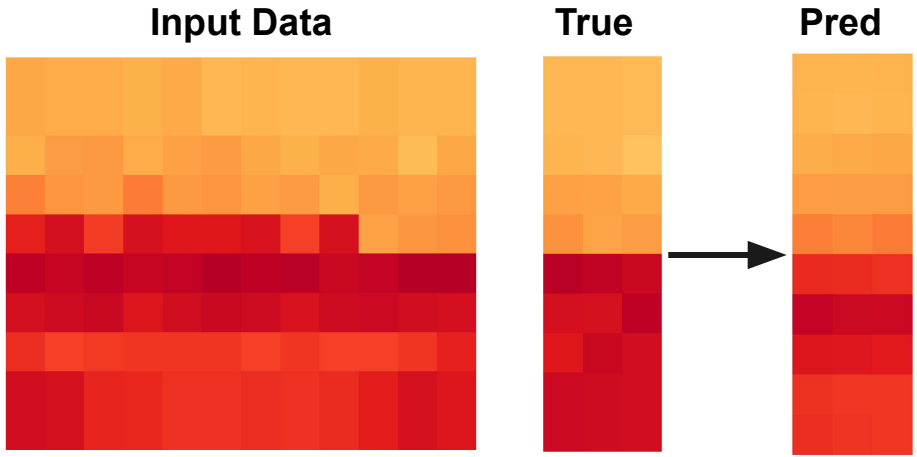
압구정로
Urban road





Study Site 4

CNN Key Takeaways

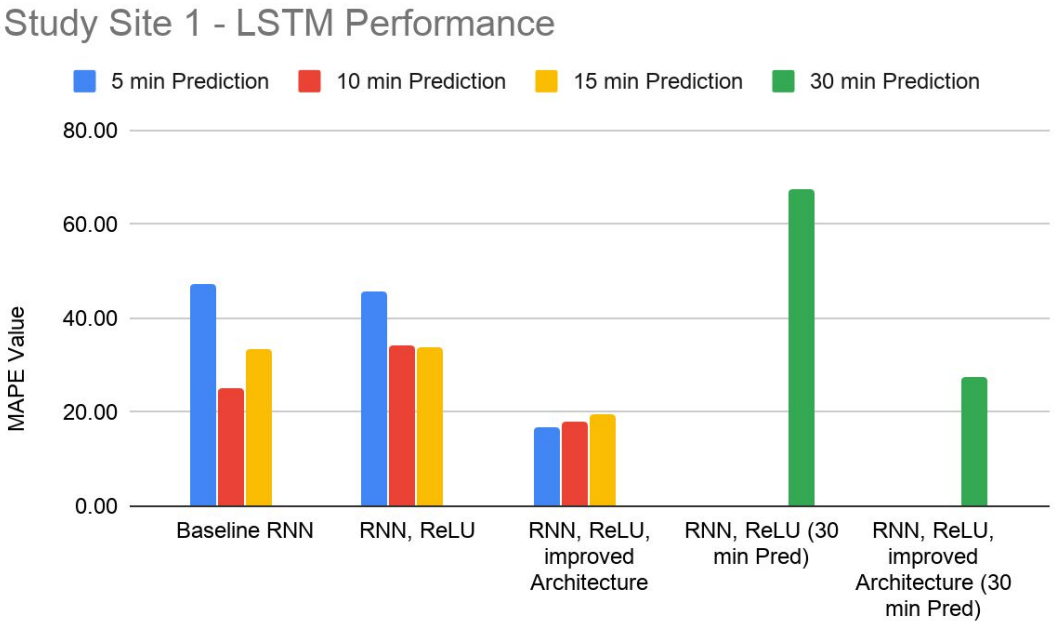


- **Typically:** many different ways to combine low-level features into higher level features
- **Here:** not many ways to combine them into higher level features

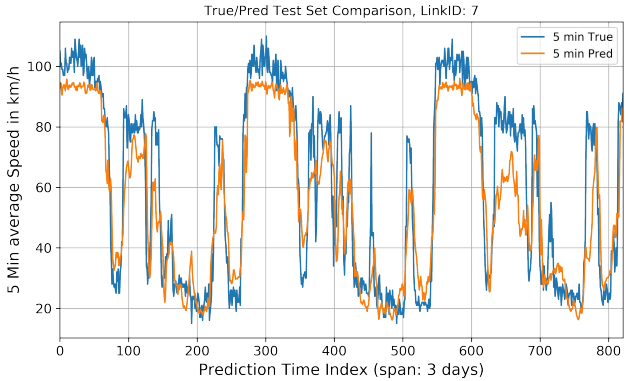
1. **Hypotheses**:
   a. Influenced by: Order of Link IDs
   b. Influenced by: Additional Channel
   c. Limited performance on more complex study sites

2. **Outcomes**:
   a. Order of Link IDs helped in Study Site 3, but not Study Site 4
   b. Added relative Speed did not increase performance much
   c. Limited performance, but still outperforming other approaches

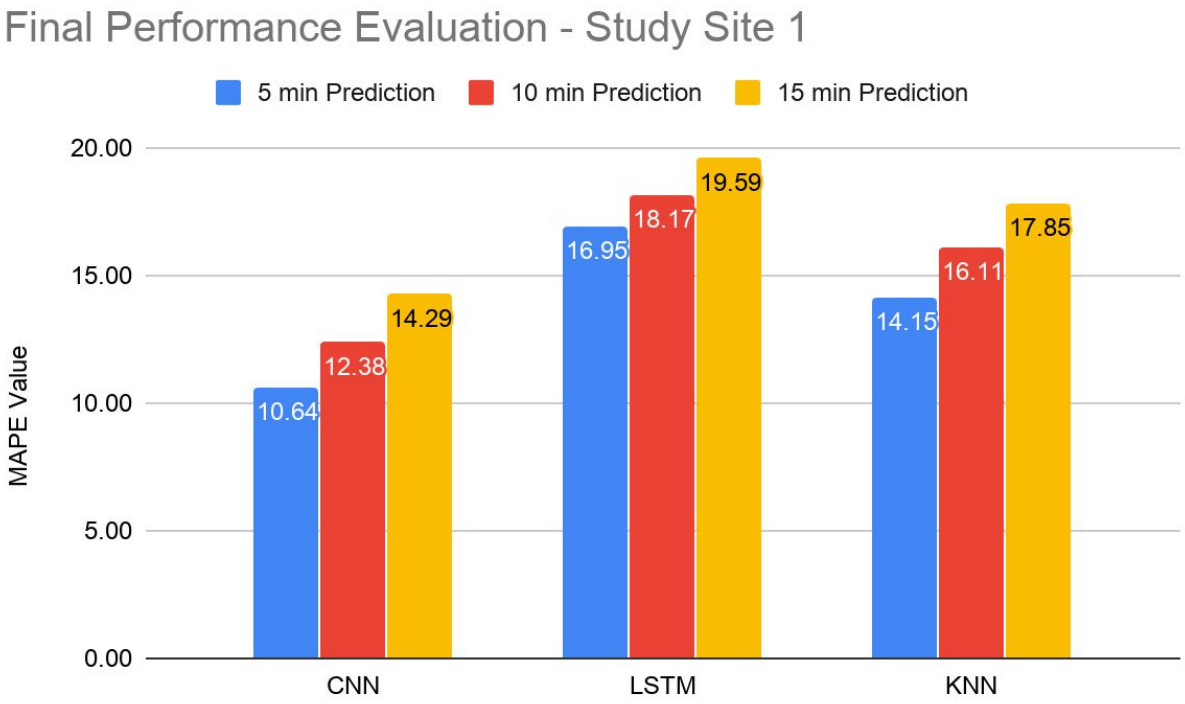
LSTM Architecture - Performance Improvement



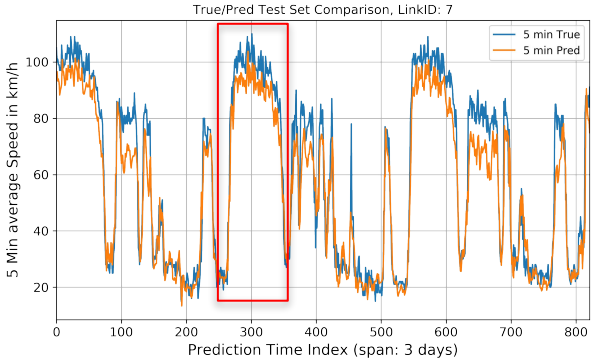

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 12, 44)            15664
_________________________________________________________________
lstm_3 (LSTM)                (None, 44)                15664
_________________________________________________________________
dense_2 (Dense)              (None, 100)               4500
_________________________________________________________________
dropout_1 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_3 (Dense)              (None, 132)               13332
=================================================================
Total params: 49,160
Trainable params: 49,160
Non-trainable params: 0
_________________________________________________________________
```



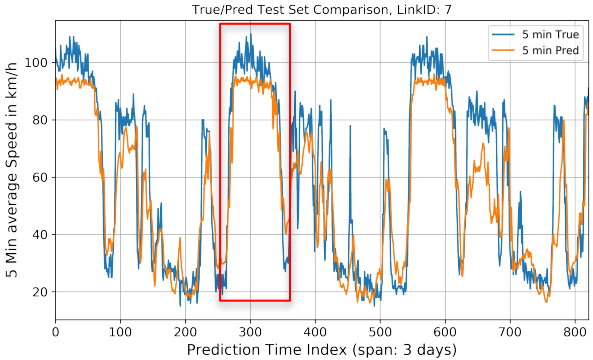
Performance Comparison - Study Site 1



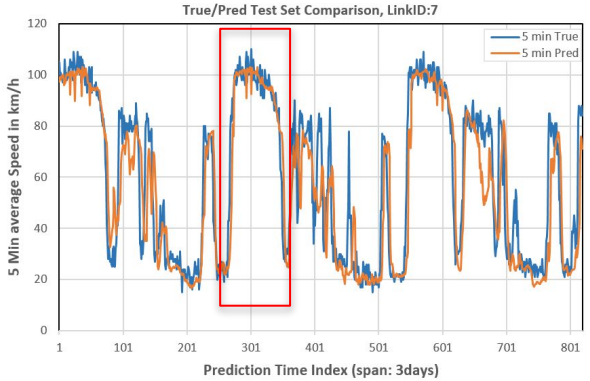
Performance Comparison - Study Site 1



CNN

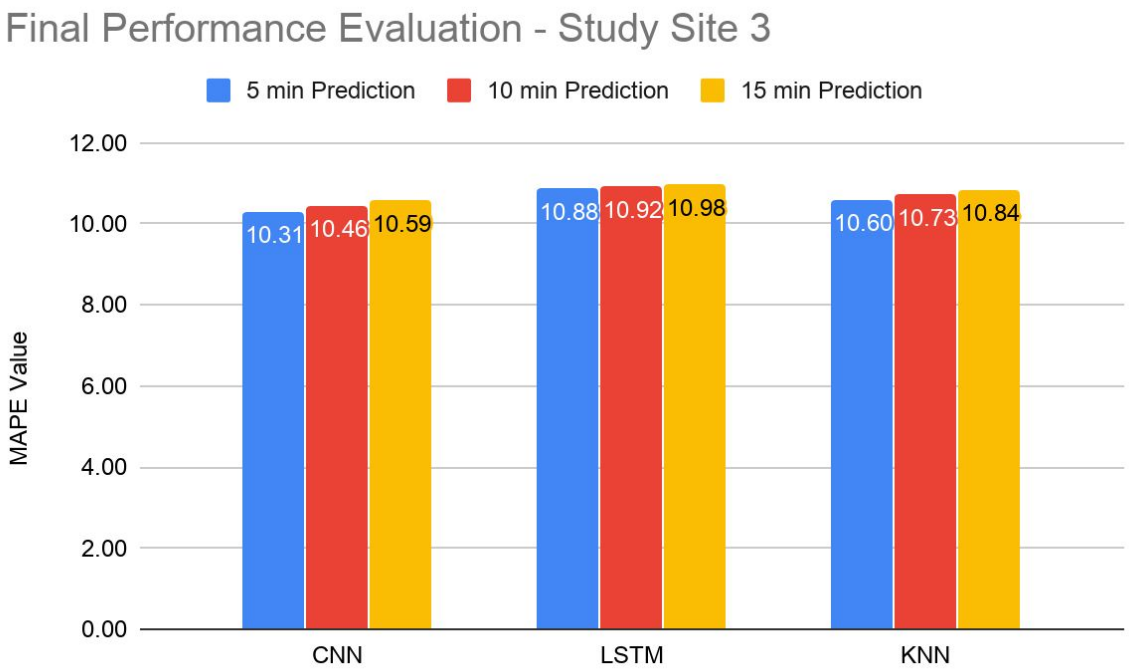


LSTM

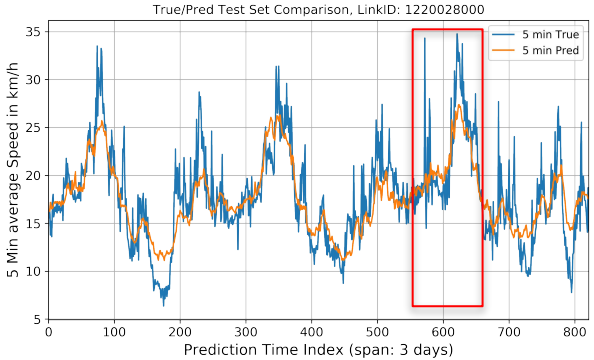


KNN

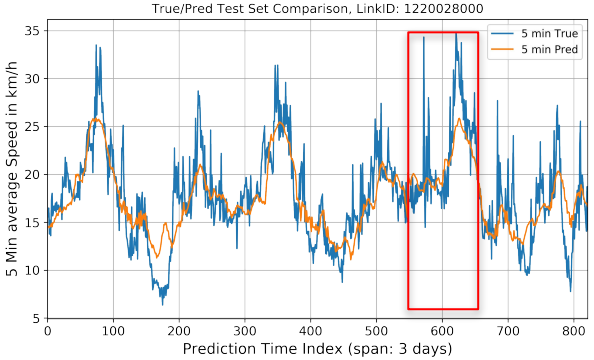
Performance Comparison - Study Site 3



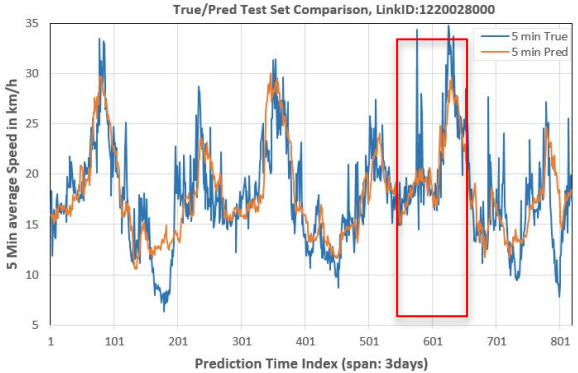
Performance Comparison - Study Site 3



CNN

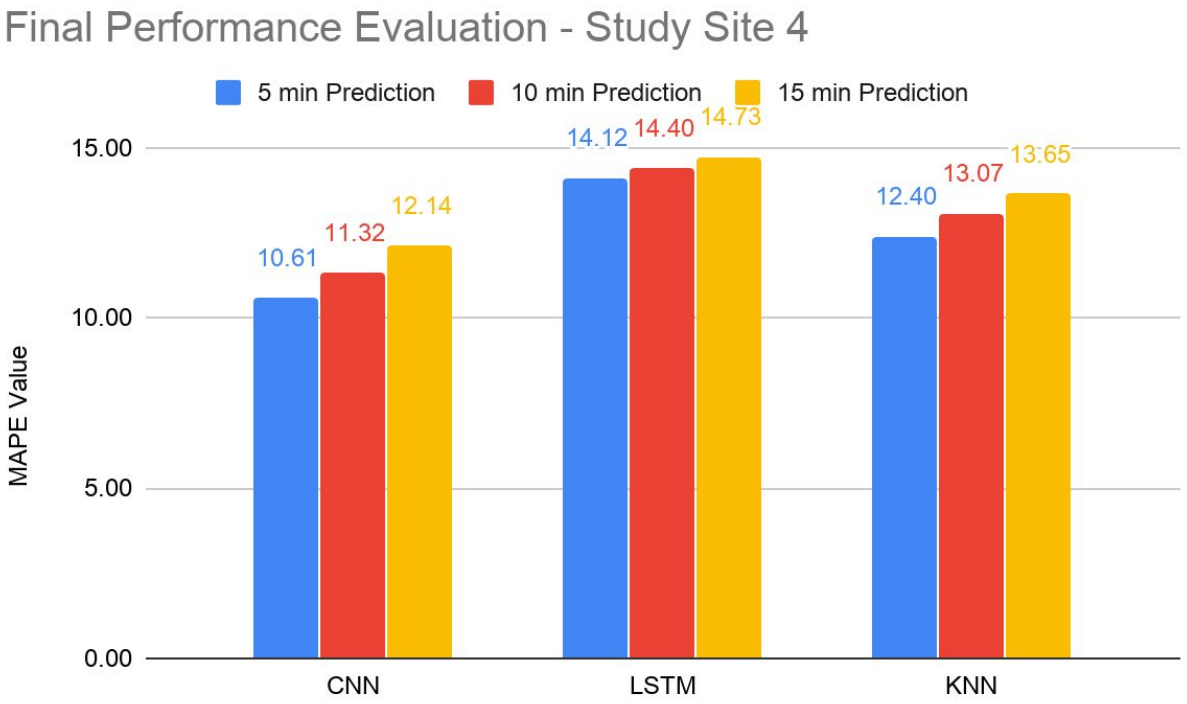


LSTM

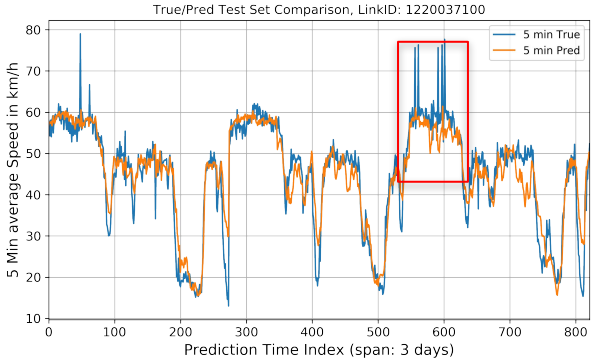


KNN

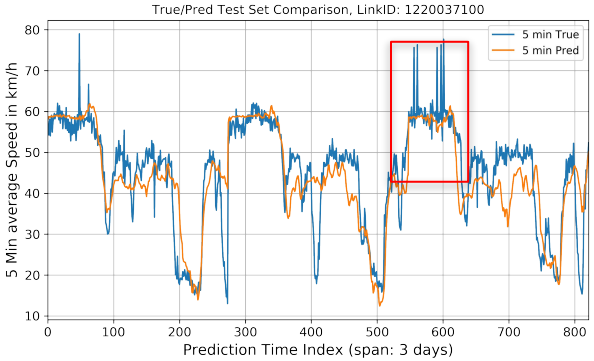
Performance Comparison - Study Site 4



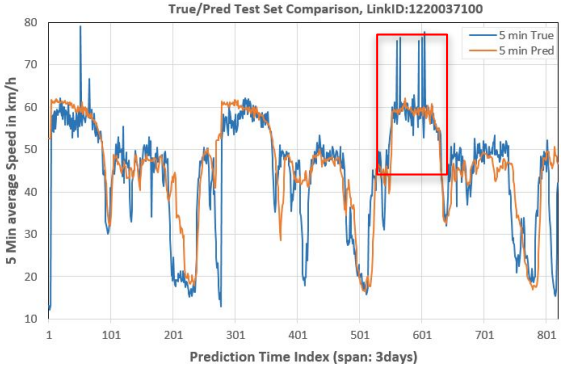
Performance Comparison - Study Site 4 (Bridge)



CNN



LSTM



KNN

Conclusion

Overall Comparison

Models	Rank	Cons	Pros
ARIMA	4	- large data set - seasonal variation	- less time consuming - white-box
KNN	2	- Cannot be trained	- Simple algorithm and intuitive
LSTM	3	- Abstracting from detailed fluctuation	- Learning general Trends
CNN	**1**	- Not beneficial for unseen situations	- Capturing detailed fluctuation

Contributions

- ✓ Extensive evaluation of various learning approaches within different geometry
- ✓ Modified and compared related works
- ✓ Analyzed performance of different approaches to understand how they are *learning* in various traffic situations

Outlook & Possible Future Work

Improvements:

- ✓ Add weekends or holidays
- ✓ Add auxiliary information (beneficial to traffic predictions)
- ✓ More complex models (e.g. Graph Convolutional Network, Attention)
- ✓ Enhance Data Foundation (add *Traffic Flow* information)

Possible Application:

- ✓ Improve Route guidance for individuals
- ✓ Provide more accurate future traffic to Traffic Management Center

Thanks for your Attention
Any Questions?

Backup

LSTM vs ARIMA model

- ✓ Performance on the amount of data..
- ✓ hyper parameter tuning for machine learning.
- ✓ uni variate vs multi variate.
- ✓ Short term forecasting and long term forecasting.
- ✓ performance on stationary and non stationary data set.
- ✓ stacking of LSTM networks.
- ✓ Poor performance of LSTM due to heavy tuning of parameters.
- ✓ VAR requires pre specified lags (by choosing model order).
- ✓ LSTM does not require pre specified window of lagged observations as input values.
- ✓ VAR training time is very short and does not require a lot of time to tune the model.

Time-Series Type	RNN Performance	Classical Model Performance
Short Time-Series	Not enough data to train.	Symbolic Regression, HMMs perform well.
Long Time-Series	Able to optimize.	Classical Model Performance is Equivalent to RNN.
Multivariate Short Time-Series	Not enough data. While RNNs able to represent any function, need a lot of data.	Multi-varaite regression, Symbolic regression, Hierarchical forecasting perform well.
Multivariate Long Time-Series	RNN is able to model nonlinear relationships among features. Computationally efficient. Automatic feature selection.	Computation efficiency may not be optimal. Feature selection challenging.