UC San Diego

DSE 260B: Group 3
Time Series Forecasting
Capstone Project, June 4th, 2021

Akash Shah, Aparna Gupta, Daniel Roten, Kevin Lane, Raul Martinez
Advisor: Dr. Rose Yu

Agenda

1. Introduction
2. Data Pipeline and Environment
3. Findings through EDA
4. Modeling methods
    a. Autoregressive Model
    b. Seq2Seq model
    c. PDE-based model
    d. DCRNN definition (Modeling Product)
5. DCRNN challenges and scaling
6. Model interpretation and model comparison
7. Visualization Dashboard
8. Demo

Team Roles and Responsibilities

Data Analysis

- **Project manager**: Kevin
- **Budget manager:** Raul
- **Record keeper:** Akash
- **Solution architect:** Daniel
- **Visualization & dashboard developer:** Aparna

Library Development

- **Project manager & integration lead:** Kevin
- **Classical time series model:** Aparna
- **Deep seq2seq:** Raul
- **Spatiotemporal forecasting:** Akash
- **PDE + deep learning:** Daniel

Why create a deep learning library for time series?

- **Why deep learning for time series?**
  - Broad application in many domains including finance, health, etc.
  - Traditional models rely on strong modeling assumptions
  - Deep learning can leverage rise in large-scale sensor data
  - Improve forecasting of multivariate data and data with spatial characteristics
- **Library Development**
  - Develop open-source deep learning library for time series forecasting in PyTorch
  - Existing libraries statsmodels and sktime are limited to traditional models
  - GluonTS is a deep learning time series forecasting library based on MXNet

https://github.com/Rose-STL-Lab/torchTS

Introduction

- Traffic patterns have changed as a result of the COVID-19 pandemic
- Models relying on historical data will perform poorly as a result
- Models that account for changing patterns (additional features, online learning, etc.) will outperform these models
- Build a deep learning library for time series forecasting

Data Sources - Traffic

Traffic - CalTrans PeMS

- Traffic observations recorded over 30 second windows
- Multiple rollups available (5 min, 1 hour, etc.)
- 40,000 sensors installed on freeways across California
- No bulk download option provided by Caltrans
- Python script using Beautiful Soup web scraper retrieves data by district level, date range, file type

- **Frequency:** 5 minute interval (new data published daily)
- **Size:** ~70 MB/day (San Diego), 12.5 MB/day (Bay Area)
- **Data Link:** PeMS Data Clearinghouse

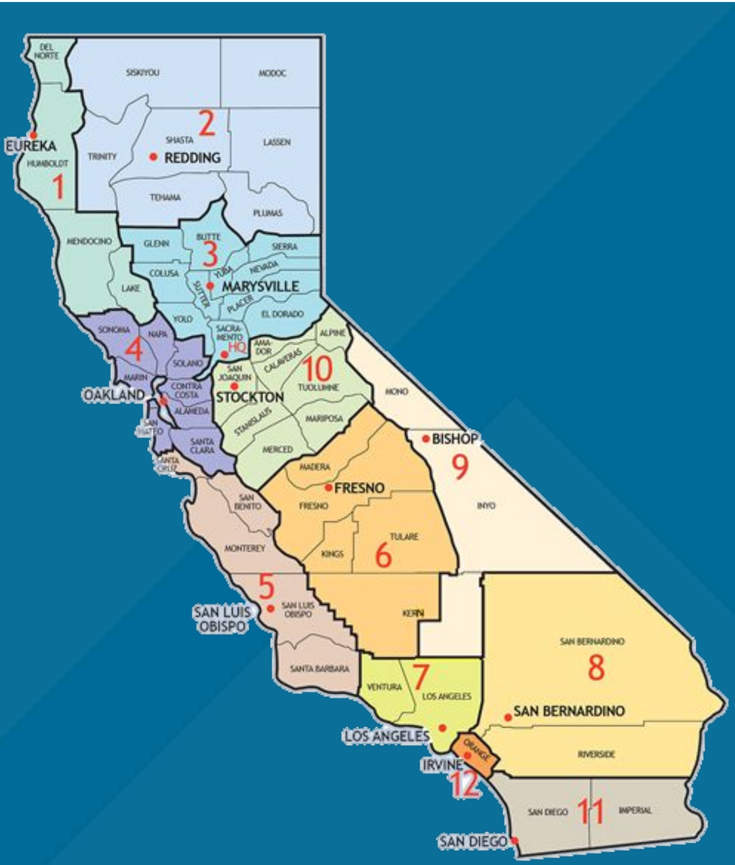


Image credit: Caltrans

Data Sources - COVID-19 and USDOT

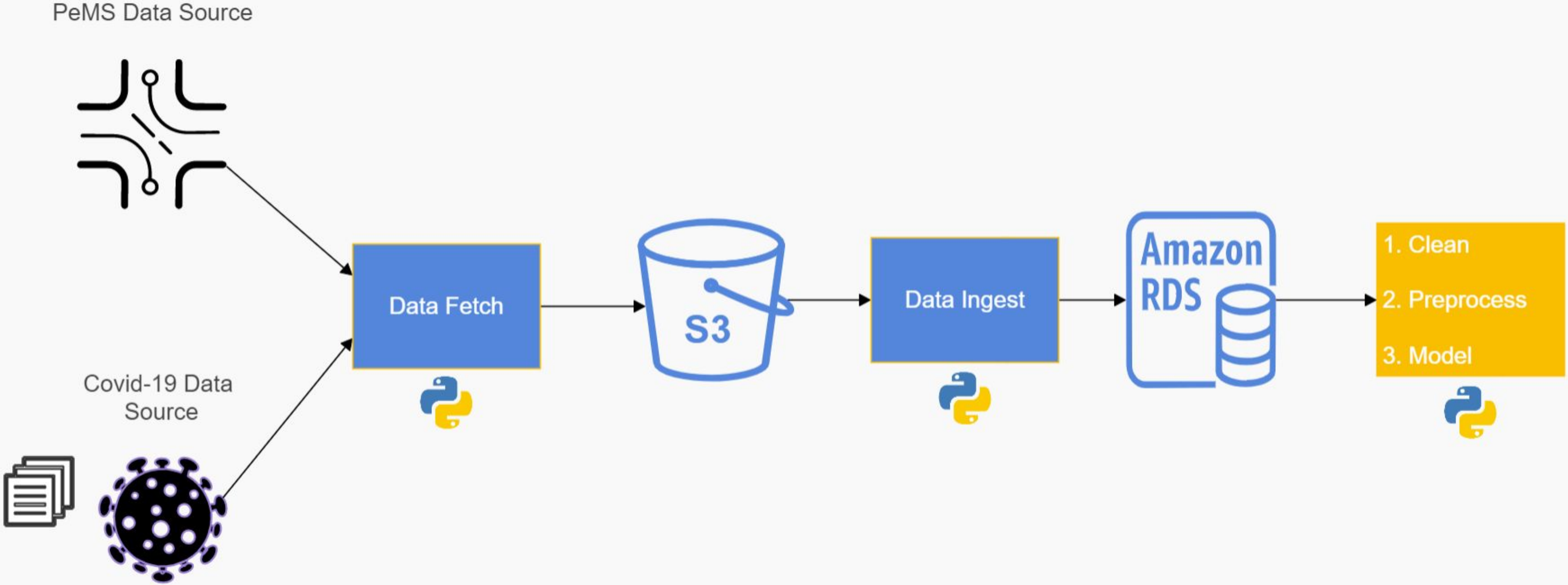
COVID-19

- U.S. county cases and deaths
- Provided by Johns Hopkins as Github repository
- git pull to refresh
- **Frequency:** Daily
- **Size:** ~7.6 MB
- **Data Link:** Johns Hopkins COVID-19 Repository

USDOT (U.S. Department of Transportation)

- Road network
- Categorical road layer data in form of shapefiles
- Provided via web interface at County level
- **Frequency:** One Time
- **Size:** ~36 MB for Santa Clara county
- **Data Link:** USDOT Tigerline Roads

Data Pipeline



Acquiring Caltrans traffic data

- Data cleaning and storage
  - Scraped Caltrans PeMS website and saved daily files to S3
  - Read raw files from S3, clean data, and insert to RDS database
- Data preparation
  - Read traffic data for stations and time period of interest from RDS
  - Prepare data for model (order stations, create sliding window, etc.)
- Available features at each station (per lane and across all lanes):
  - Metadata (ID, location, freeway number/direction, postmile)
  - Timestamp
  - “Total flow” (number of vehicles)
  - Average speed
  - Average occupancy

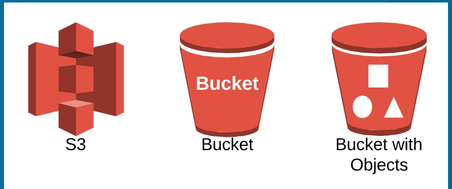


Image credit: Amazon Web Services, PostgreSQL

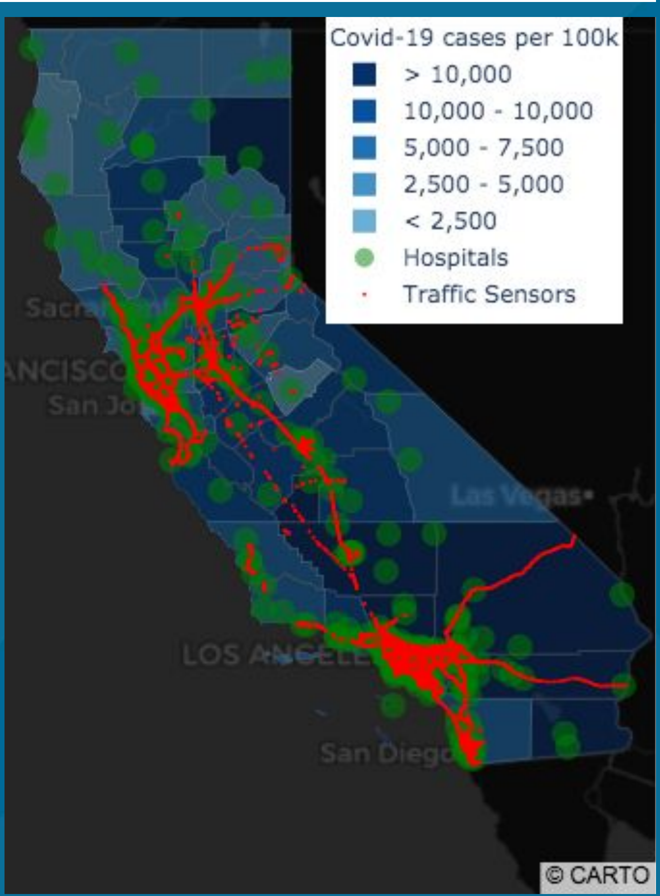
Acquiring COVID-19 cases

Data cleaning and storage

- Directly cloned from Johns Hopkins Github repository
- Stored on AWS S3

Available features at each COVID-19 reporting location

- Metadata (UID, coordinates, county name, FIPS, population)
- Datestamp
- Number of confirmed cases, number of fatalities



Data Environment

- Amazon Web Services (AWS)
  - Simple Storage Service (S3)
  - Relational Database Service (RDS)
  - Elastic Compute Cloud (EC2)
- PostgreSQL
- Open-source data sources
- Nautilus cluster - Kubernetes deployments

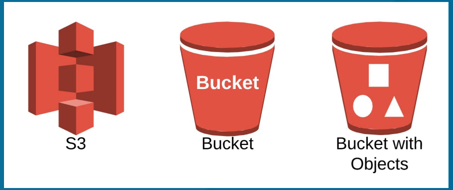
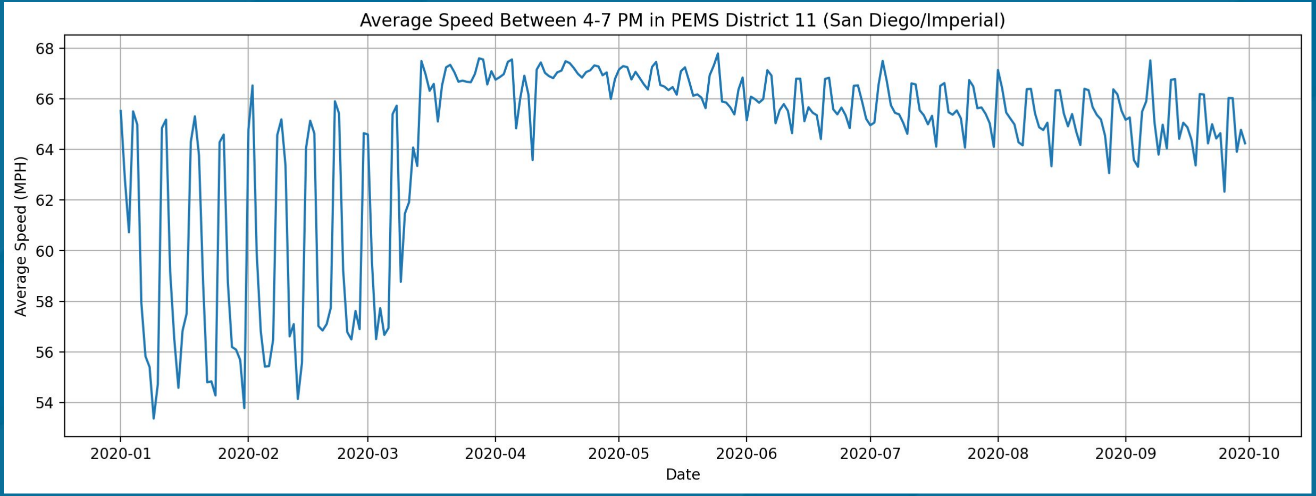


Image credit: Amazon Web Services, Kubernetes, PostgreSQL

Intro: Problem Definition

- Traffic is noticeably lighter during the COVID pandemic
  - Increase in average speed
  - Decrease in amplitude between weekday/weekend fluctuations
- Can we build a traffic forecasting model that is sensitive to external factors?
- Combine with spatiotemporal characteristics of traffic data

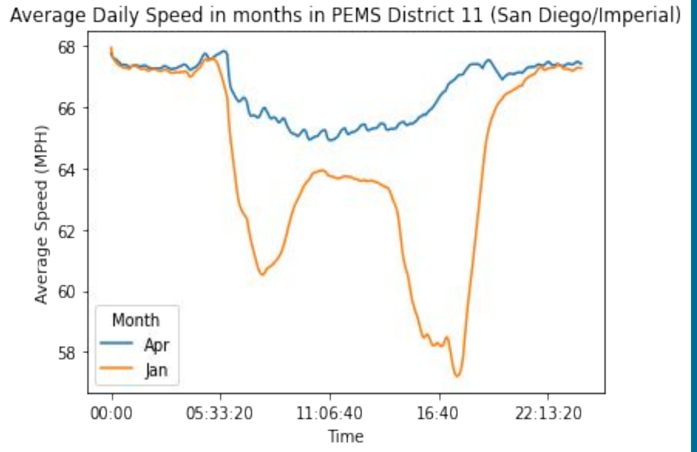


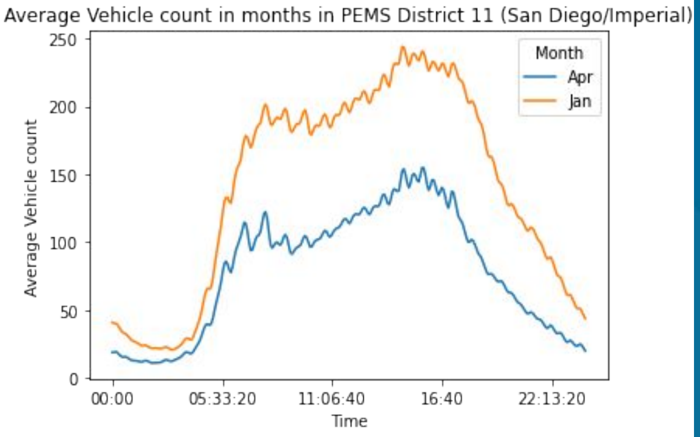
Findings through EDA: COVID-19 changed traffic patterns

EDA Findings on the Traffic Dataset dataset during 2020:

- Top figure Shows “Average Daily Speed” (vehicles/5 min) over the course of an average day for the same time frames. It shows a dramatic reduction in average speed variation.

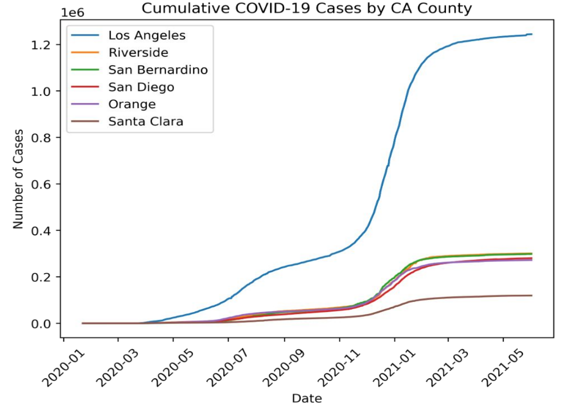
- Bottom figure Shows “total flow” (vehicles/5 min) over the course of an average day for the same time frames. It conveys a large decrease in the number of vehicles on the road, particularly after roughly 5:30 AM (left).

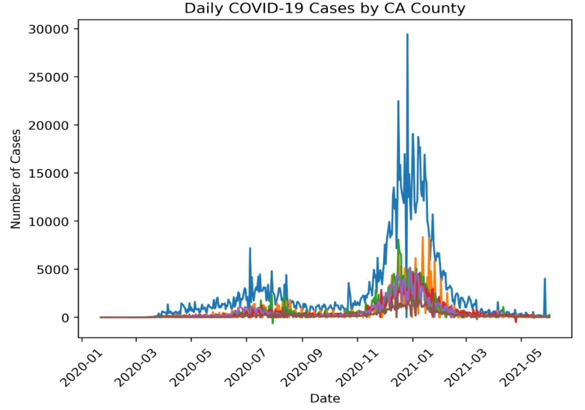


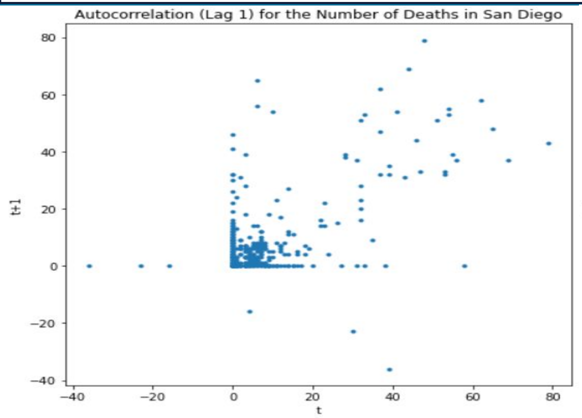


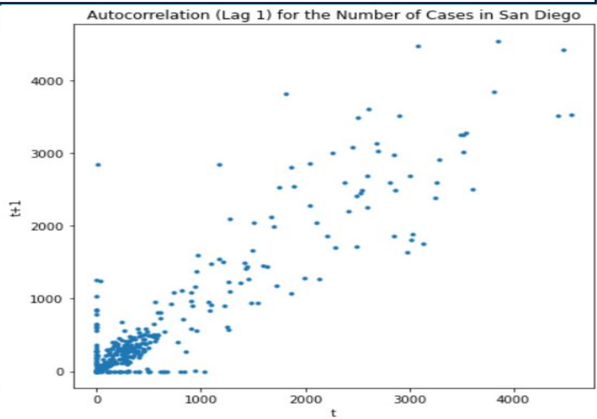
Findings through EDA: COVID-19 COVID Cases by County

- COVID-19 Cases Trend in California (Top 6 Counties by Total Cases)
- New COVID cases steadily increased till Aug with a subsequent dip
- Cases spike again starting Oct, reaching an all time high between Dec 2020 and Jan 2021
- Autocorrelation for the number of COVID Deaths (bottom left) and COVID Cases (bottom right) in San Diego


Cumulative COVID-19 Cases by CA County


Daily COVID-19 Cases by CA County


Autocorrelation (Lag 1) for the Number of Deaths in San Diego


Autocorrelation (Lag 1) for the Number of Cases in San Diego

Proposed Solution and Approach

Approach for forecasting COVID-19’s impact on Traffic

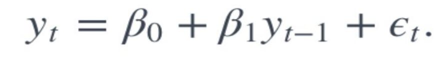
- Gather, clean and store traffic and COVID-19 data into Amazon RDS.
- Represent data as graphs or adjacency matrices to incorporate spatial information from external factors.
- Implement below forecasting methods and evaluate how including COVID-19 data at county level and spatial information improves traffic predictions around certain areas.

Approach for Deep Learning Library - TorchTS

- Classical Time series (AR,MA,ARIMA)
- Seq2Seq methods (Encoder - Decoder)
- Spatiotemporal methods - Graph Convolutional Networks (GCNs)
- Partial Differential Equation based Deep Learning.

Autoregressive - Classic Time Series Model

- The autoregression (AR) method models the next step in the sequence as a **linear function** of the observations at prior time steps.



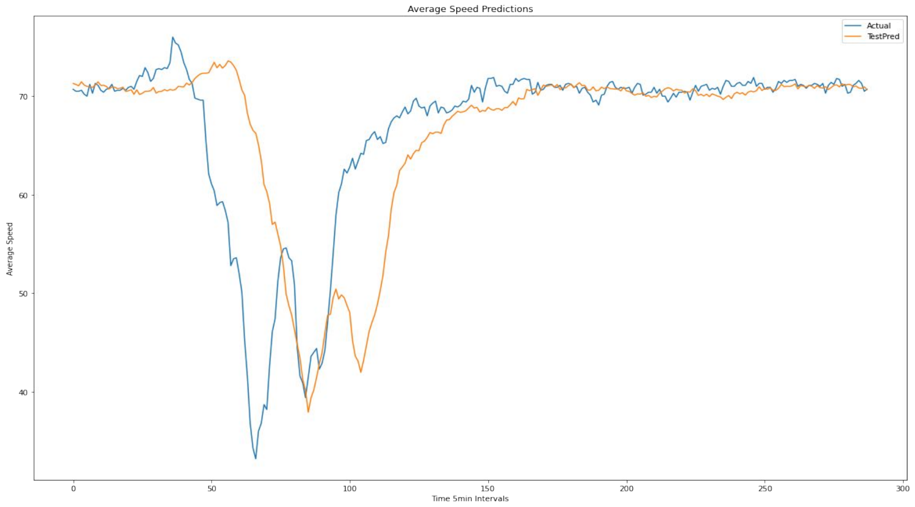
$$y_t = \beta_0 + \beta_1 y_{t-1} + \epsilon_t .$$

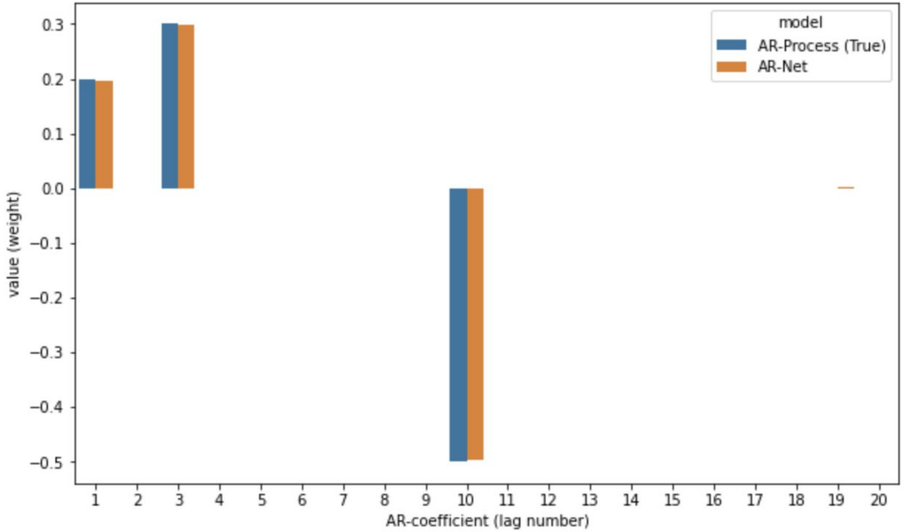
- The AR model created has been trained using the traffic sensor measurements at five min. intervals for 320 stations in the Bay areas for the time period Jan’20 to Jun’20. The Mean Absolute Error calculated for 1 hour horizon is 2.2905

- The dataset contains each input as a scalar value of 12 values(lag) representing the average speed of the vehicles in 5 min interval, depicting the 60 min lag.

Autoregressive - Performance and Results

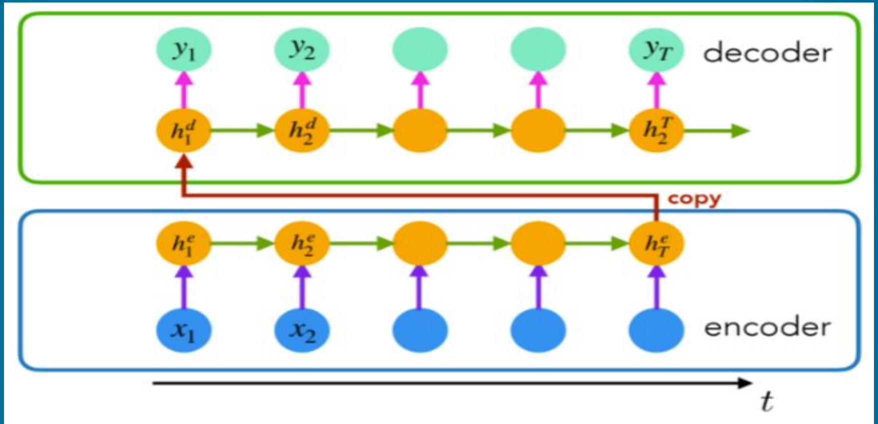
- Autoregressive neural network implementation is inspired by AR-Net paper
- **Left:** AR(12) model successfully predicts next traffic measurement
- **Right:** AR(20) model fit on AR(3) process correctly determines model coefficients

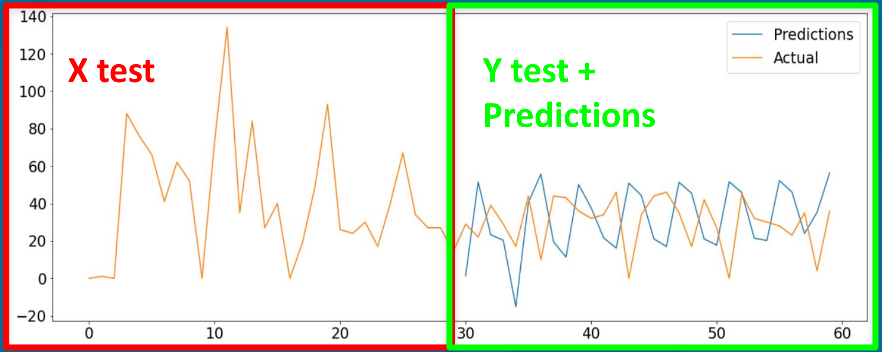




Seq2Seq model

- Maps the input sequence to a fixed-sized vector with an encoder to the target sequence with a decoder.
- RNNs are used to retain the sequential information in the time series.
- **LSTM** (Long Short Term Memory): Designed for problems with long term dependencies, addresses the vanishing gradients issue.





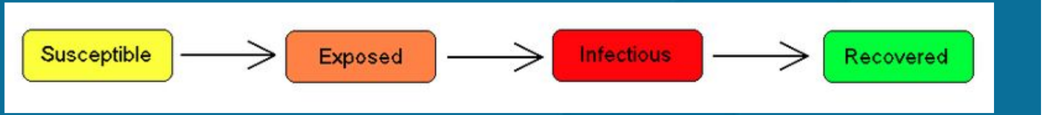
X test

Y test + Predictions

Physics-Informed COVID-19 Prediction

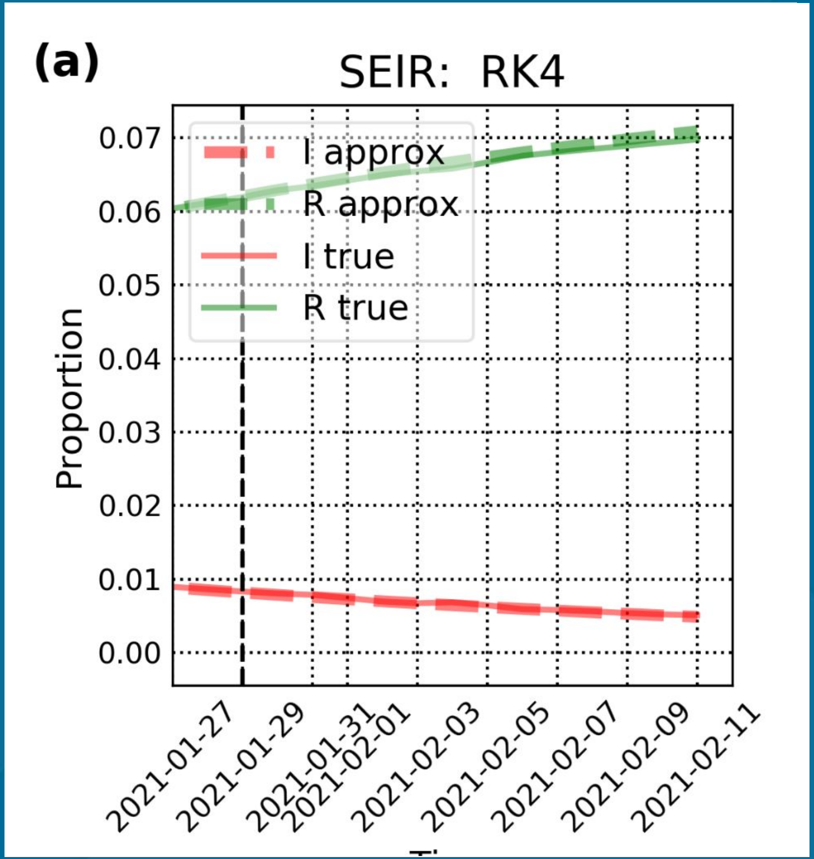
ODE-based neural network:

- COVID-19 forecasting using compartmental model:



$$\frac{dS_t}{dt} = -\frac{\beta I_t S_t}{N}$$
$$\frac{dE_t}{dt} = \frac{\beta I_t S_t}{N} - \sigma E_t$$
$$\frac{dI_t}{dt} = \sigma E_t - \gamma I_t$$
$$\frac{dR_t}{dt} = \gamma I_t$$

- Discretize in time, integrate numerically (RK4)
- Optimize values for β, γ, σ using Adam



Physics-Informed COVID-19 Prediction

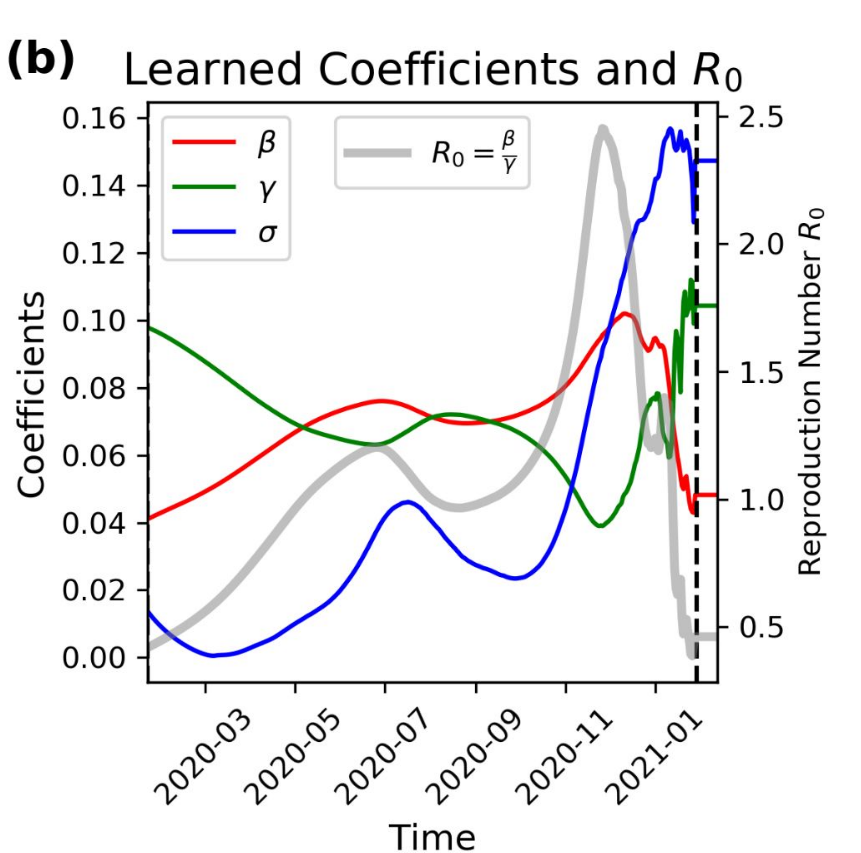
ODE/PDE-based neural networks

- COVID-19 forecasting using compartmental model:

- The basic reproduction number can be derived from contact rate β and recovery rate γ:

$$R_0 = \frac{\beta}{\gamma}$$

- R_0 was elevated during two waves with peaks in July and December 2020, but decreased sharply in early 2021



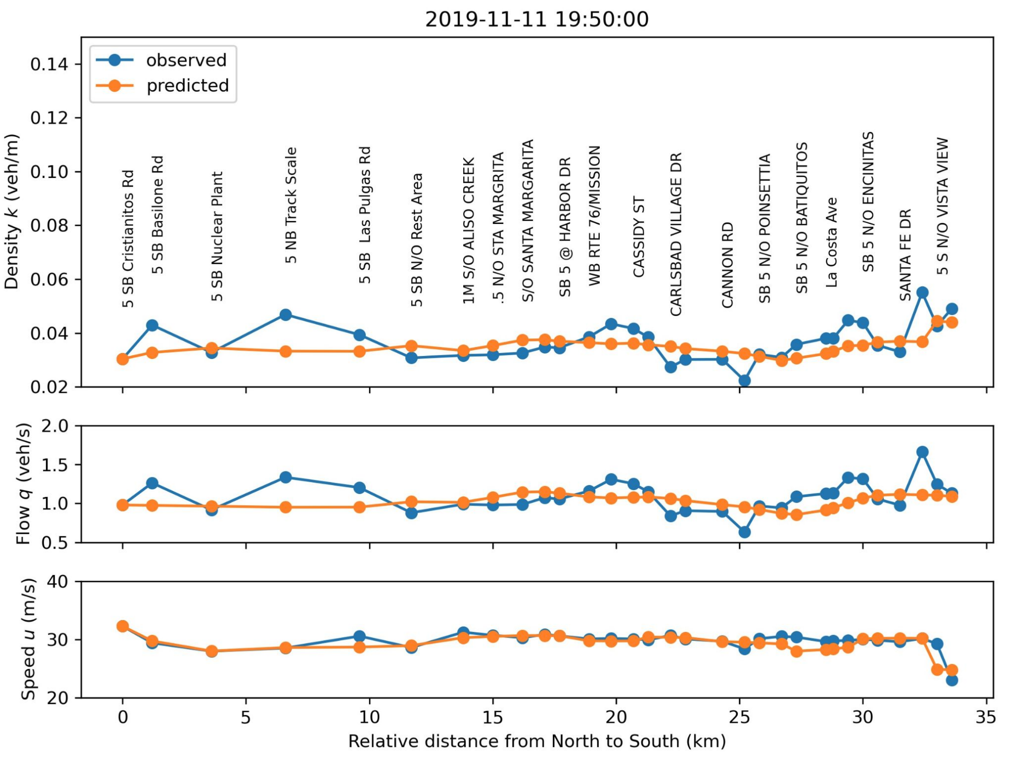
Physics-Informed Traffic Prediction

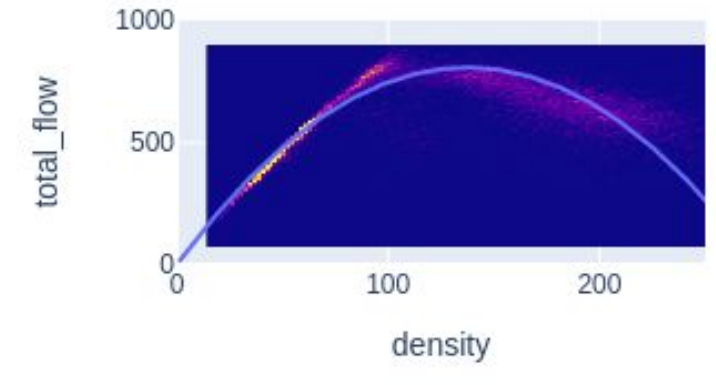
Numerically solves Lighthill-Whitham-Richards macroscopic traffic model for jam density k_j and free velocity v_f.

$$\frac{\delta q}{\delta x} + \frac{\delta k}{\delta t} = 0$$

$$q = kv$$

$$v = v_f \frac{1 - k}{k_j}$$





DCRNN (Modeling Product)

- Diffusion Convolutional Recurrent Neural Network
- Network of 320 traffic stations represented as an adjacency matrix.
- Events at a given location propagate downstream
- Graph convolutional RNN architecture
  - Spatial Dynamics - Diffused Convolutions on Graphs
  - Temporal- Stacked RNN (GRU)

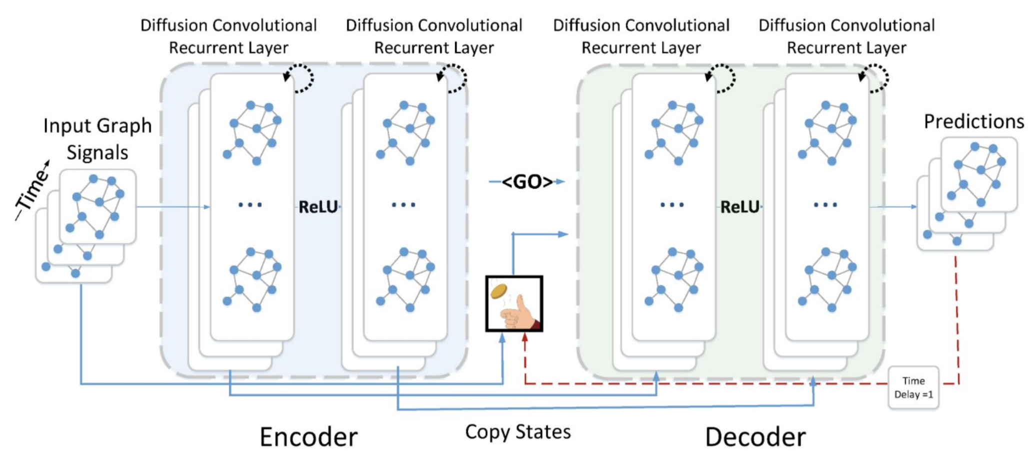

Diffusion Convolutional Recurrent Layer
Diffusion Convolutional Recurrent Layer
Diffusion Convolutional Recurrent Layer
Diffusion Convolutional Recurrent Layer
Input Graph Signals
Time
ReLU
<GO>
ReLU
Predictions
Time Delay =1
Encoder
Copy States
Decoder

Image credit: Li, Yaguang, et al. “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting.” *ICLR*, 2018.

Model Hyperparameters

~300,000 model parameters

PARAMETER	DESCRIPTION	VALUE USED
Batch size	Size of input values used in one forward pass	64
Filter type	***The type of Graph Convolution***	***Dual Random Walk***
Number of RNN layers	***The number of stacked RNN layers to use***	***2***
Maximum diffusion step	***Maximum steps of random walks***	***K = 3***
Optimizer	Optimizers used to calculate Gradient Descent	Adam
Learning Rate	Rate at which the takes optimizer takes steps	0.01
Curriculum Learning	Method to learn an encoder decoder network	True
Scheduled sampling rate	Probability of row being ground truth or prediction	0

Model Training challenges

- Computational Challenges
  - Long training times and heavy computational resources needed
  - Resolved using larger AWS resources g4.dn xlarge and the Nautilus cluster for batch jobs
- Model training challenges
  - Loss value: 0.0000
  - NULL Values in the data points at certain stations
  - Resolved using rolling average imputation

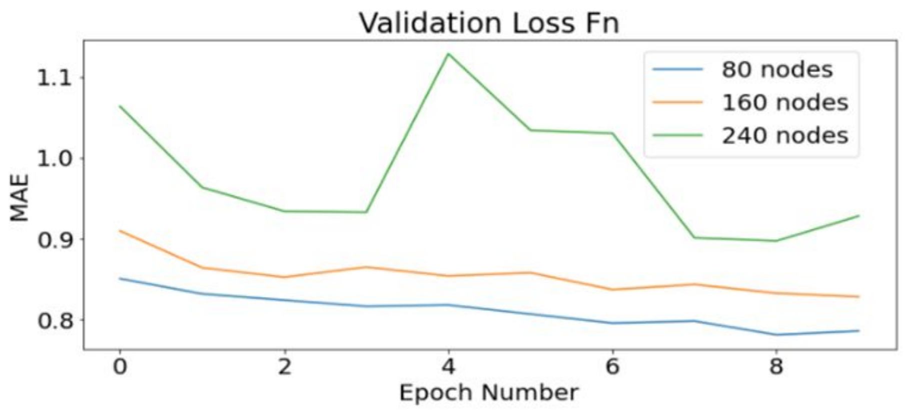
	Seq2seq	**Spatio-Temporal**
Recurrent Neural Network (RNN) Mechanism	Long Short-Term Memory (LSTM)	Gated Recurrent Unit (GRU)
Number of Parameters	399,617	372,353
Estimated number of training hours	1.5 hr/epoch	1 hr/epoch
Execution Time (original EC2 threshold)	~150 hrs	~100 hrs

Image credit: Amazon Web Services, Kubernetes

Scalability

Data Scalability

- Varying dataset sizes by using different subsamples of the traffic graph, 25%, 50%, and 75%.
- MAE increases as we increase the number of nodes

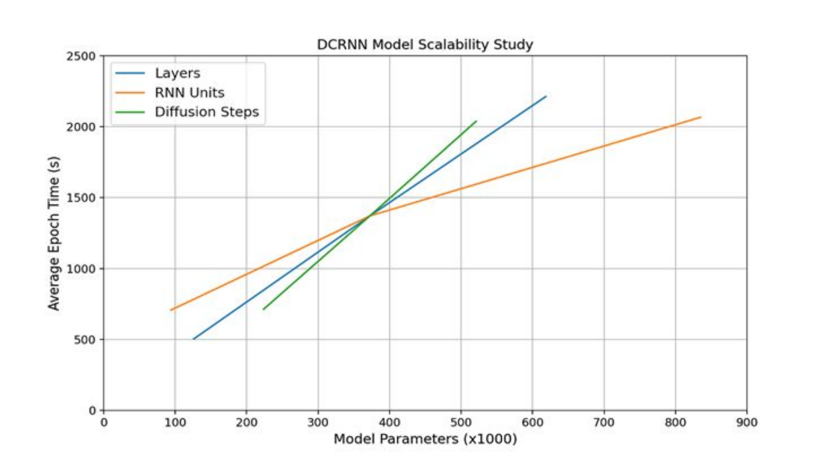


Model Scalability

- Compute time for different architectural changes
- Average Epoch time increases with increase in model parameters

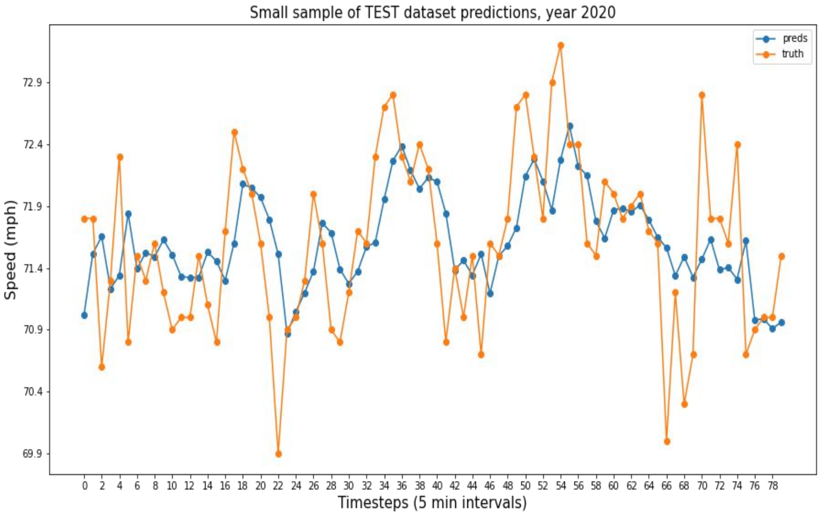
Compute Scalability

- Wrap the library around PyTorch Lightning library
- Reduction in training time , from 1.4 hours to 4 minutes

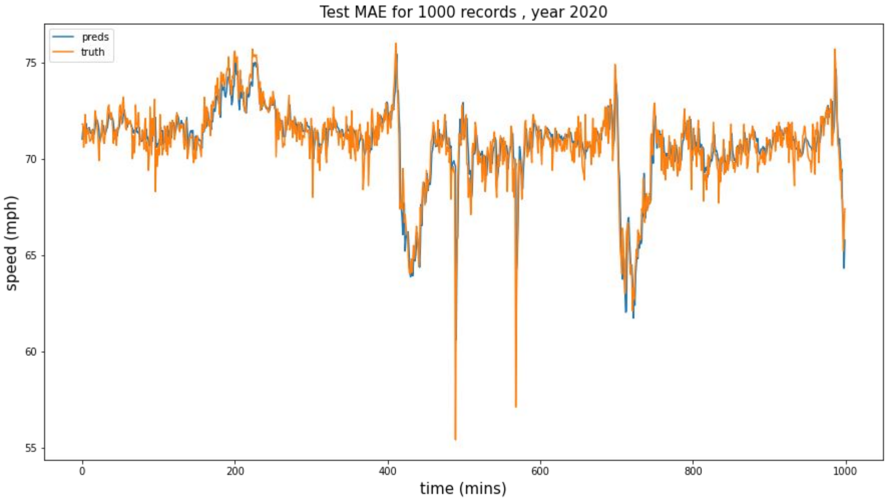


Model Interpretation

- MAE of 1.02 across all of the 320 sensors for the test set.
- Lower than the MAE reported in the paper for 2017 traffic data.
- Attributed to simpler data patterns and also a robust DCRNN model

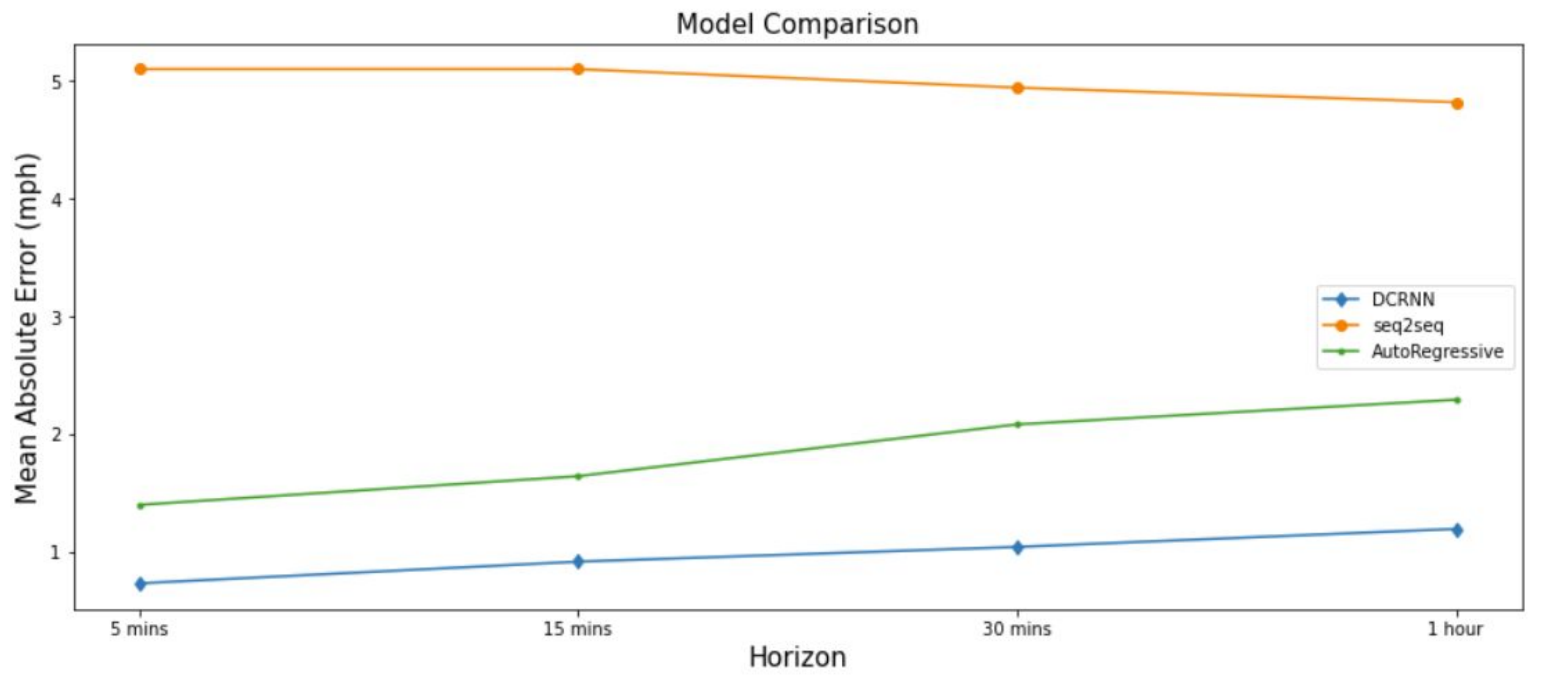


Small sample of TEST dataset predictions, year 2020



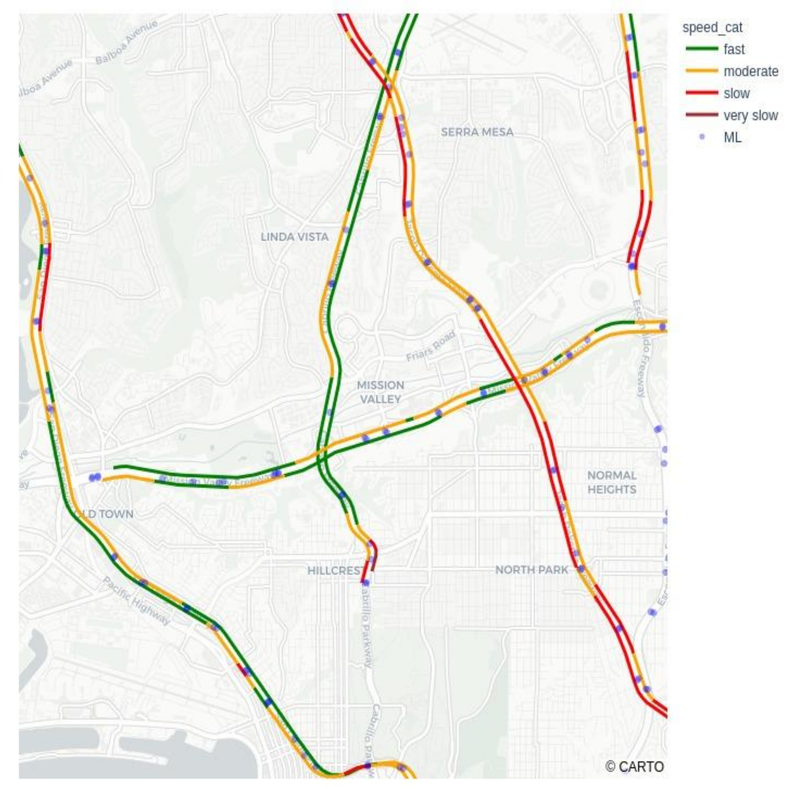
Test MAE for 1000 records , year 2020

Model Comparison



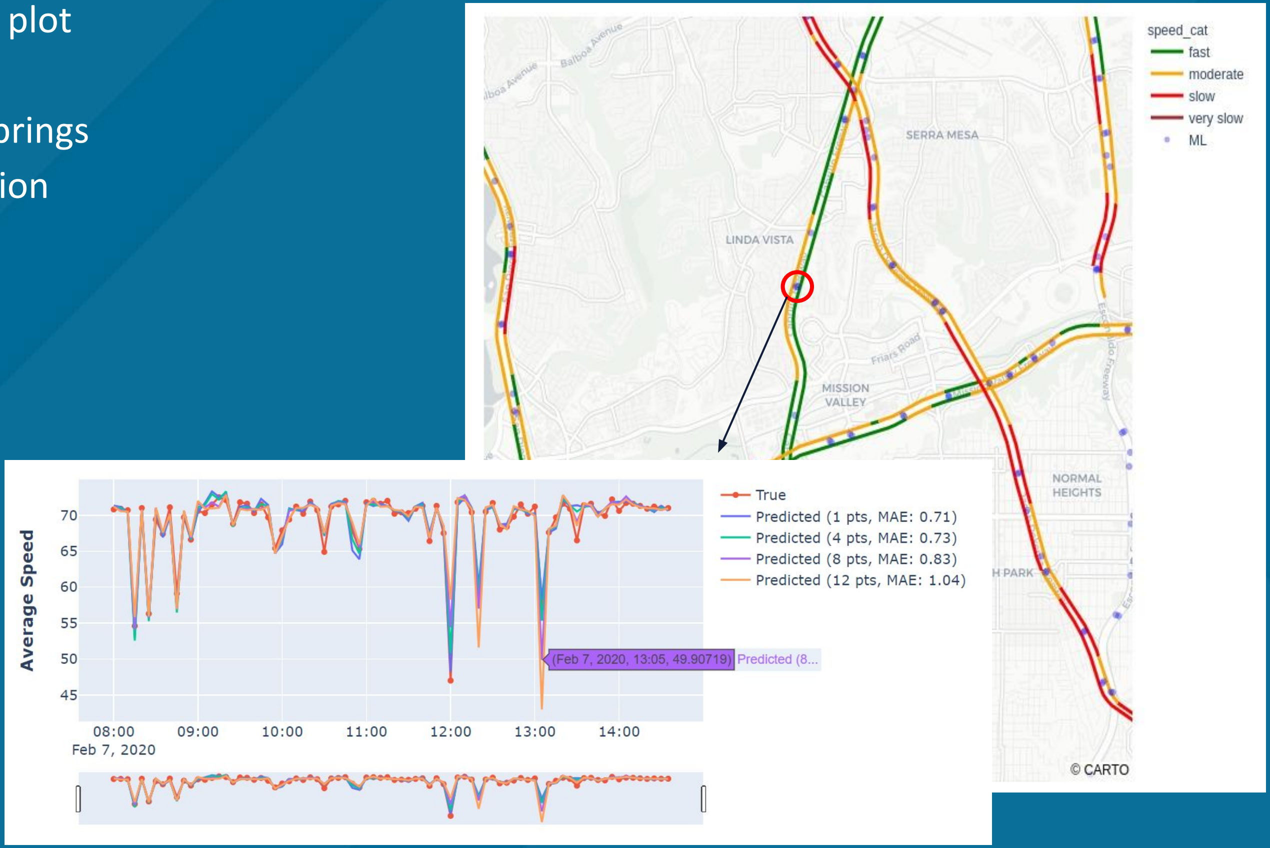
High level traffic predictions for the general public

- Draw inspiration from Google Maps
  - Ubiquitous route planning service
  - Color speed by categorical encoding
  - Provides familiar view for users
- Displays forecasted instead of current traffic
- Forecasts based on “current” traffic instead of historical averages
  - Not currently connected to live traffic data



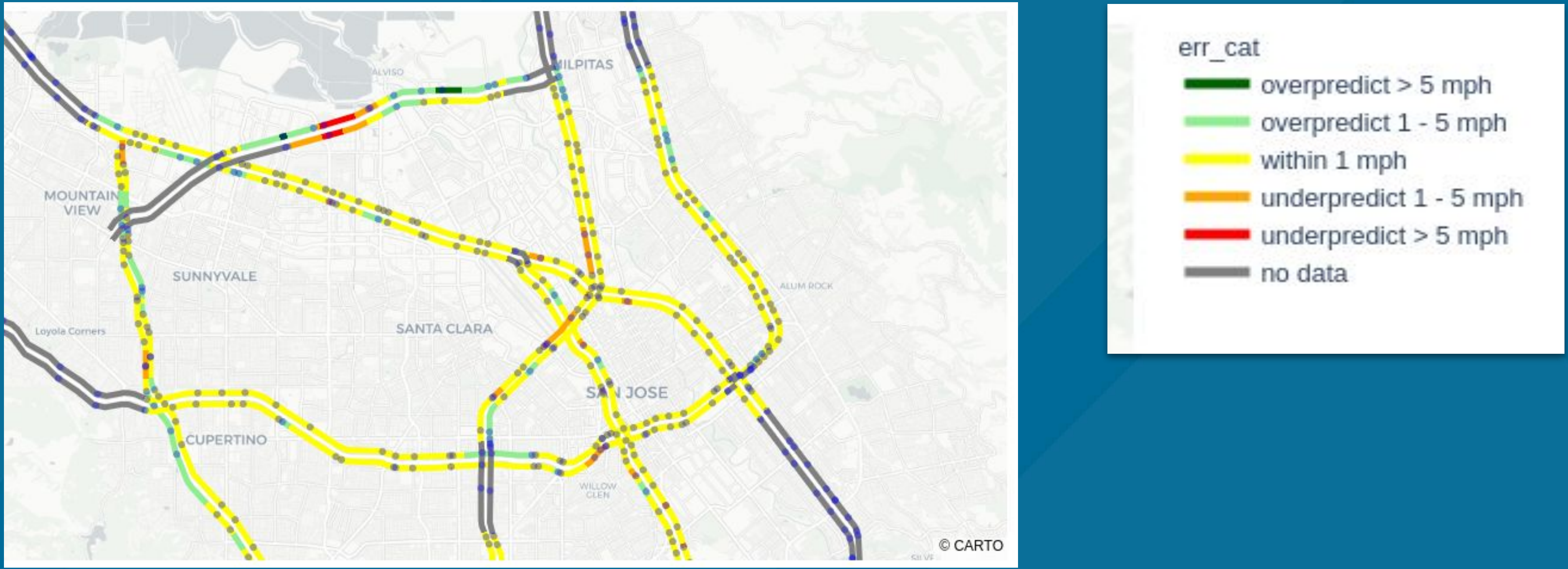
Detailed traffic predictions for traffic engineers & data scientists

- User provided with scatter plot of clickable traffic stations
- Clicking a station location brings up time series for that station
- Time series controls
  - Date Range Picker
  - Range slider
  - Multi-drop down for horizon selection



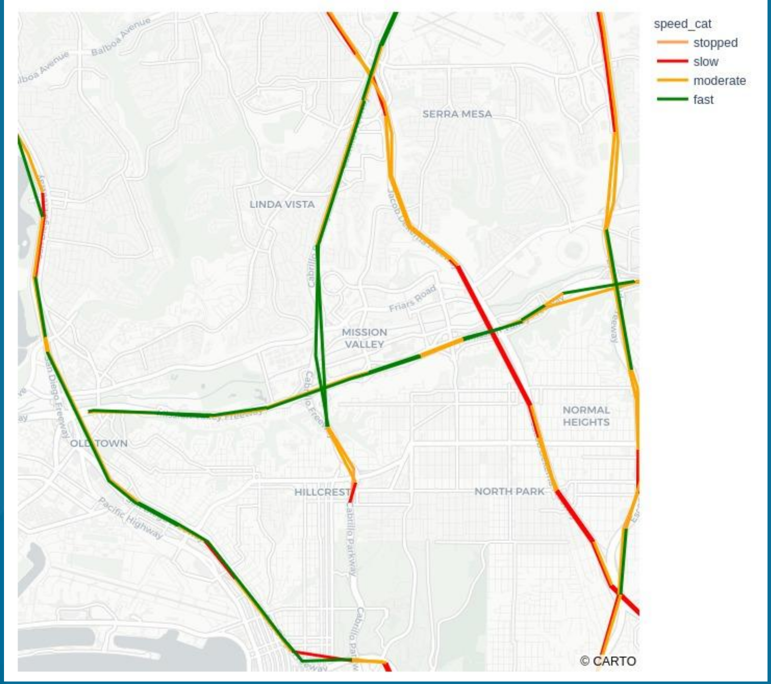
Detailed traffic predictions for traffic engineers & data scientists

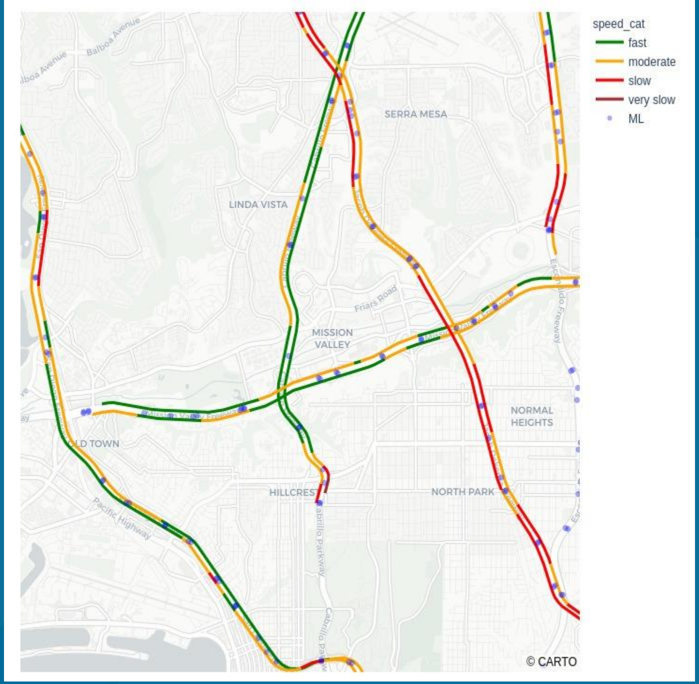
- Color freeways by average error for the selected period
- Identify regions where model underperforms relative to rest of network
- Select corresponding stations to investigate further



Refine spatial distribution for smoother visuals

- PeMS stations are rather coarse
  - Connecting with straight lines does not follow freeways
  - Unable to distinguish between freeway directions
- Interpolate onto finer coordinates obtained from Department of Transportation





Demo

Questions?