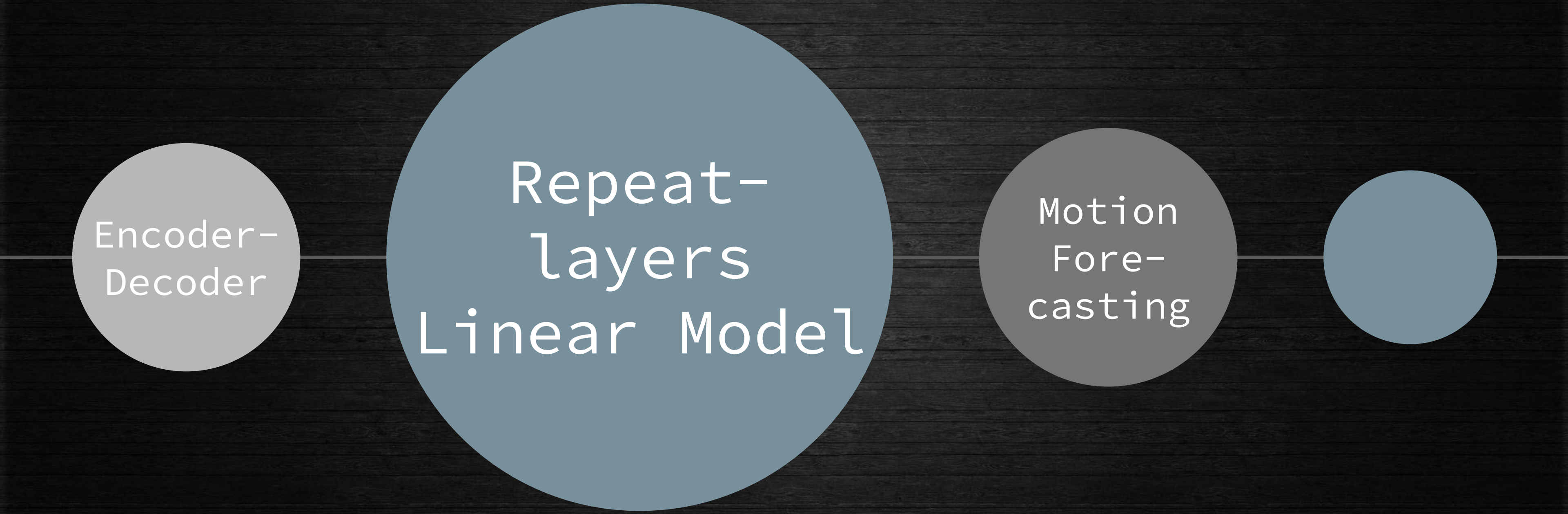
Final Presentation

- Team 2333

Summary

- Three members of our team: Felix Zhang, Manxin Zhang, Ruojia Tao
- We use a linear model to solve the problem!
- Lesson: Simpler models could perform better than complex models.

Key points



Encoder-Decoder

Repeat-layers Linear Model

Motion Forecasting

Introduction

Team Members:

Felix Zhang

- Junior double major in CSE and COGS.
- Focus on machine learning and computer graphics

Manxin Zhang

- Senior majored in Computer Science

Ruojia Tao

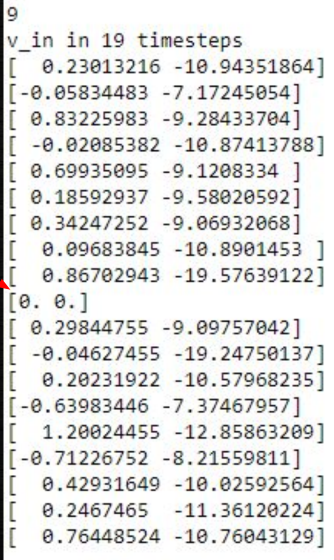
- Junior majored in Data Science

Methodology

Data Analyzing

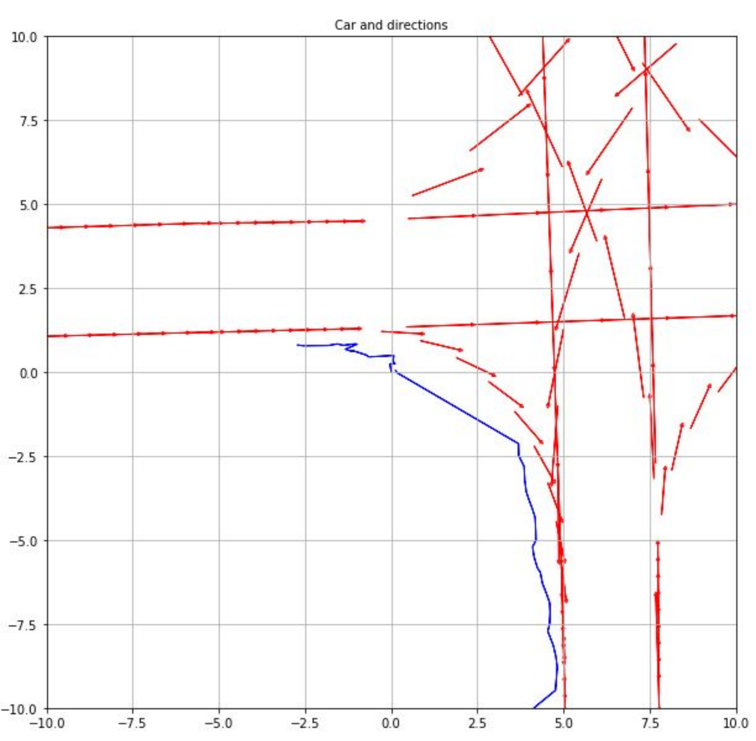
The dataset contains ERRORS!

V_in contains zeros



```
9
v_in in 19 timesteps
[  0.23013216 -10.94351864]
[-0.05834483 -7.17245054]
[ 0.83225983 -9.28433704]
[ -0.02085382 -10.87413788]
[ 0.69935095 -9.1208334 ]
[ 0.18592937 -9.58020592]
[ 0.34247252 -9.06932068]
[  0.09683845 -10.8901453 ]
[  0.86702943 -19.57639122]
[0. 0.]
[ 0.29844755 -9.09757042]
[ -0.04627455 -19.24750137]
[  0.20231922 -10.57968235]
[-0.63983446 -7.37467957]
[  1.20024455 -12.85863209]
[-0.71226752 -8.21559811]
[  0.42931649 -10.02592564]
[  0.2467465  -11.36120224]
[  0.76448524 -10.76043129]
```

Trajectories have errors

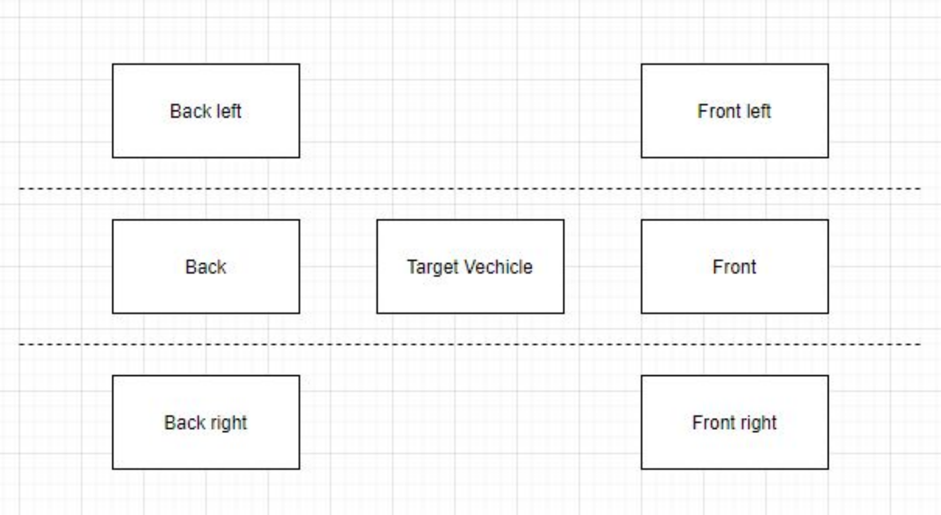


Data Processing

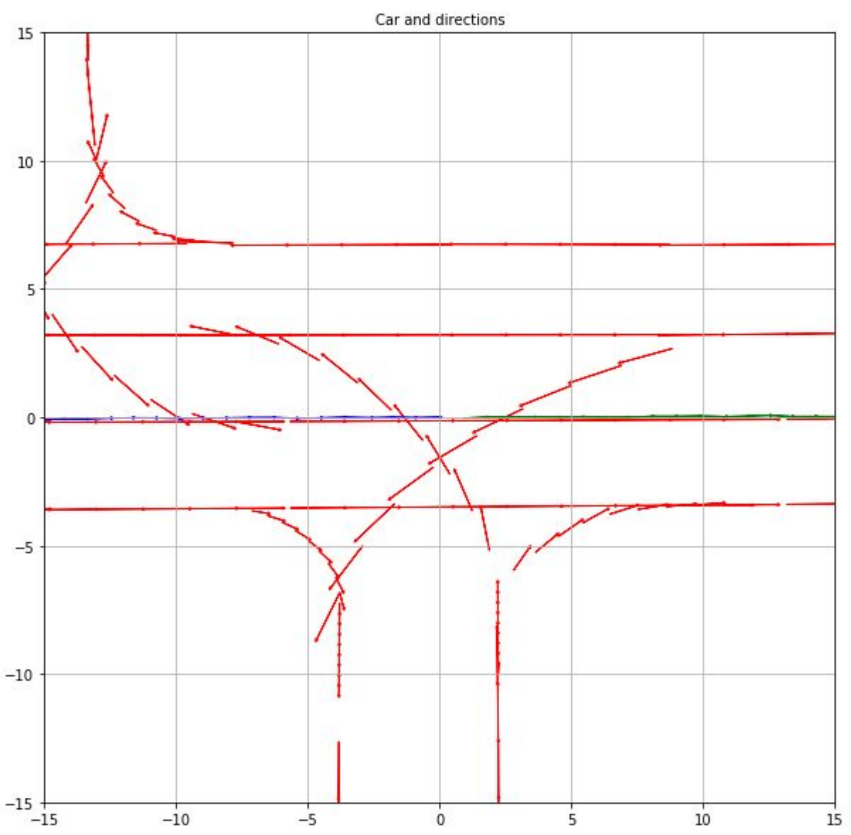
Data processing is important for NN models!

Good data in => Good output out.

Include six vehicle around the target



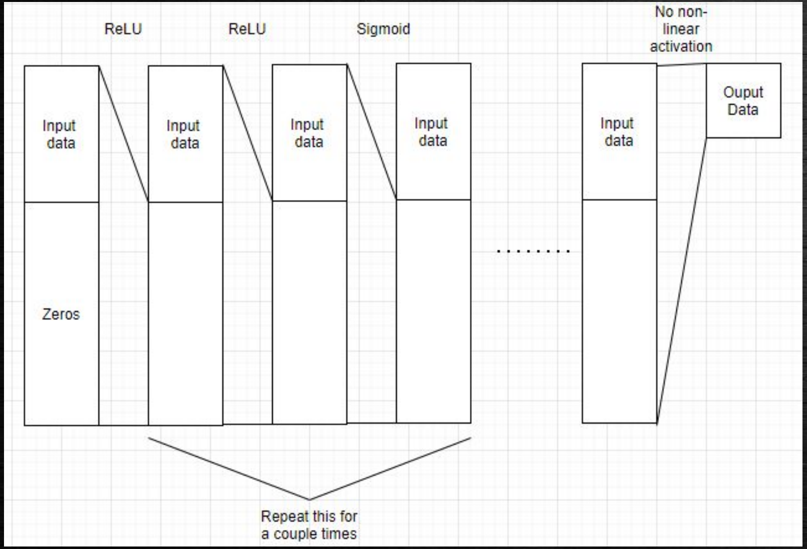
Normalization: Use relative position and make the velocity direction (1, 0)

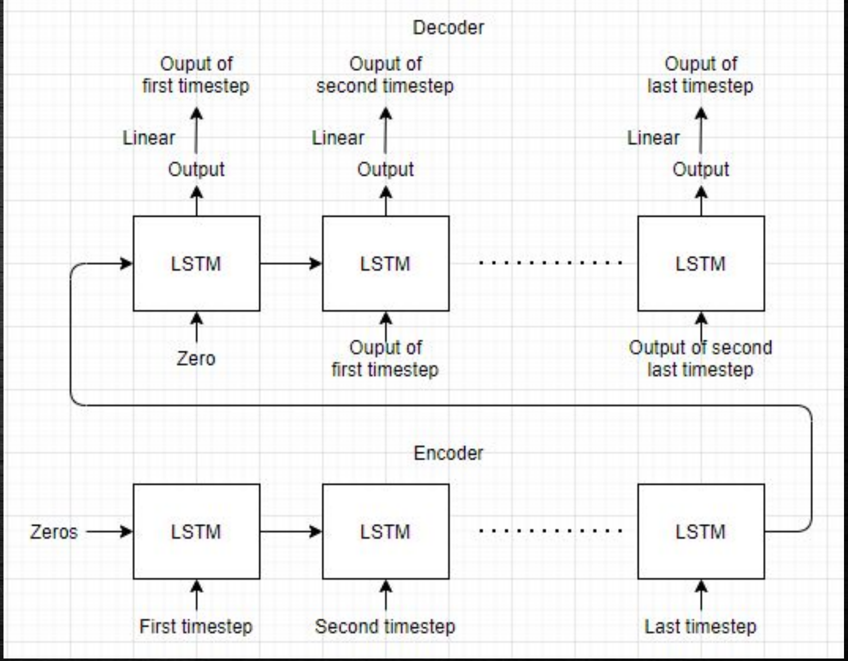


Deep Learning Model

Encoder-Decoder model

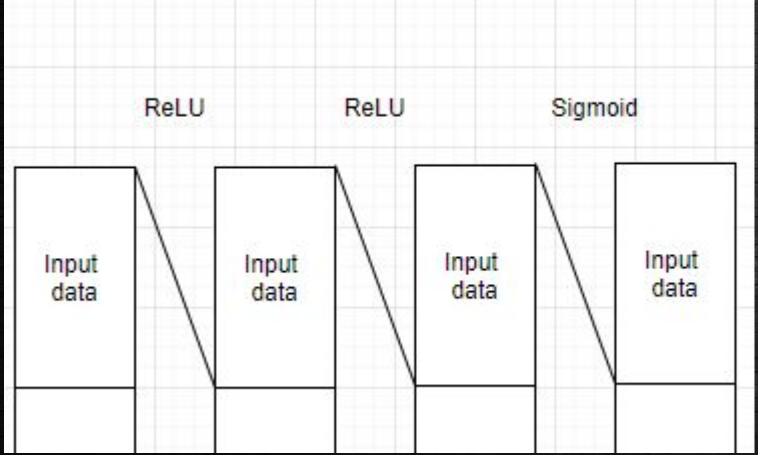
Repeat-layers linear model





Engineering Tricks

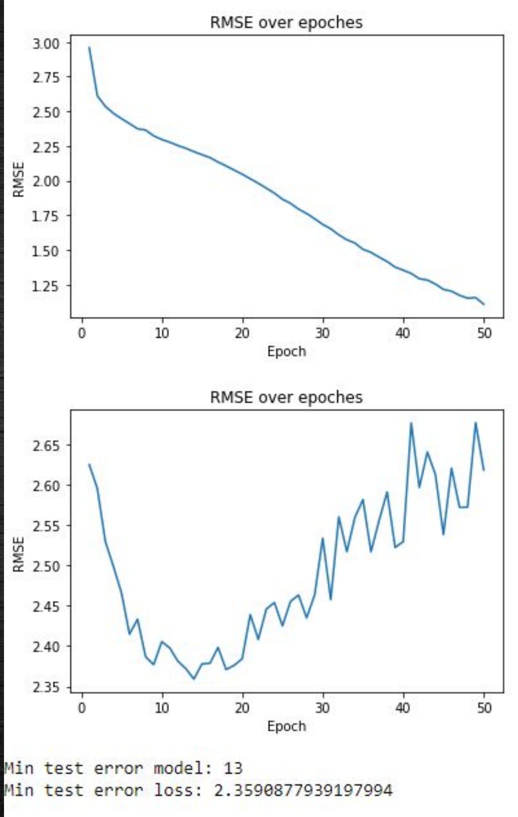
Deal with vanishing gradient



Save model after each epoch and use the epoch with lowest validation error to compute the test data.

Train Error

Test Error

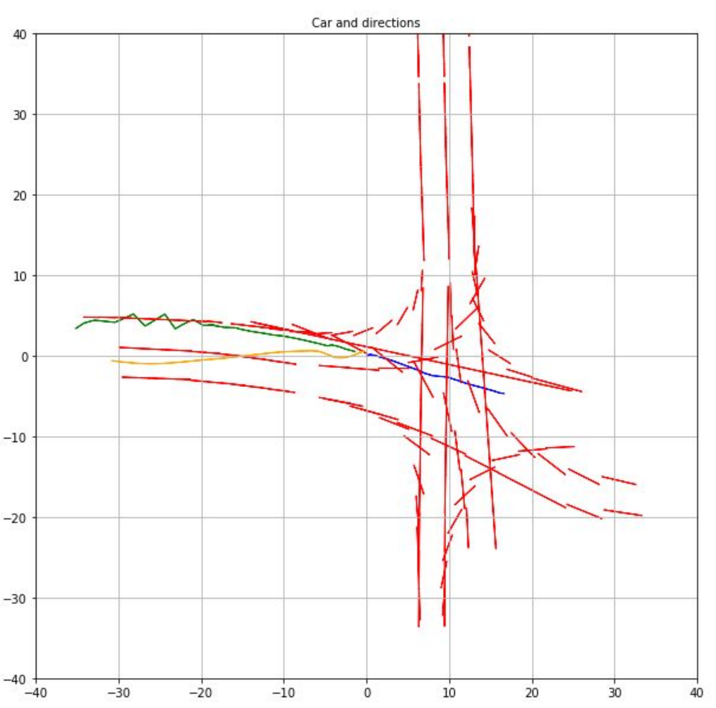


```
Min test error model: 13
Min test error loss: 2.3590877939197994
```

Experiments

Encoder-Decoder model

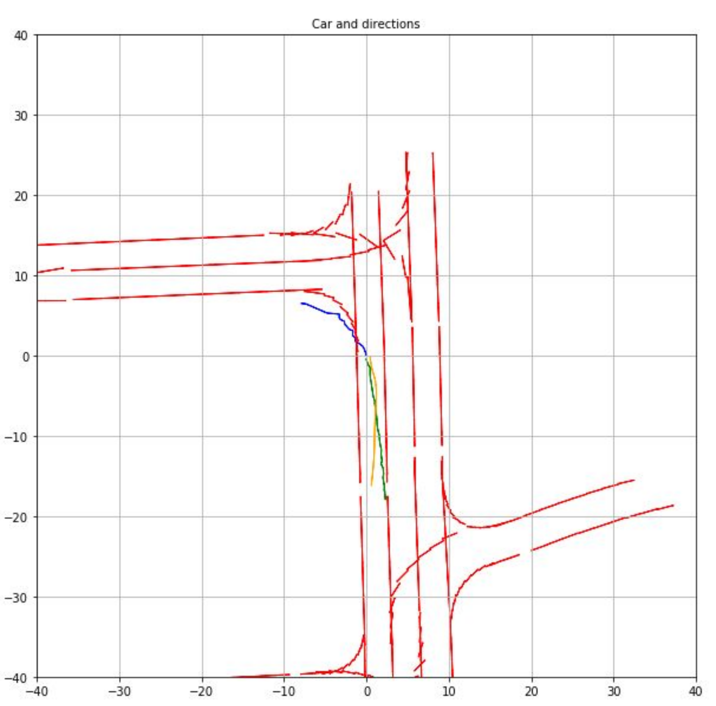
Outcome is disrupted by the last p_in/v_in



Result of encoder-decoder model

Training	Validation	Testing
2.23225	2.33789	2.22255

Can handle turning very well



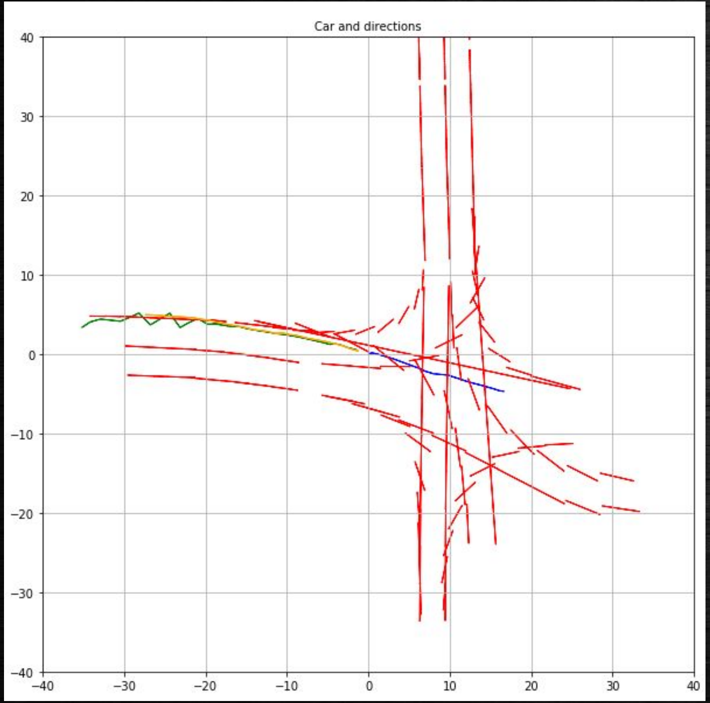
Note: blue is p_in, green is p_out, yellow is our output

Repeat-layers linear model

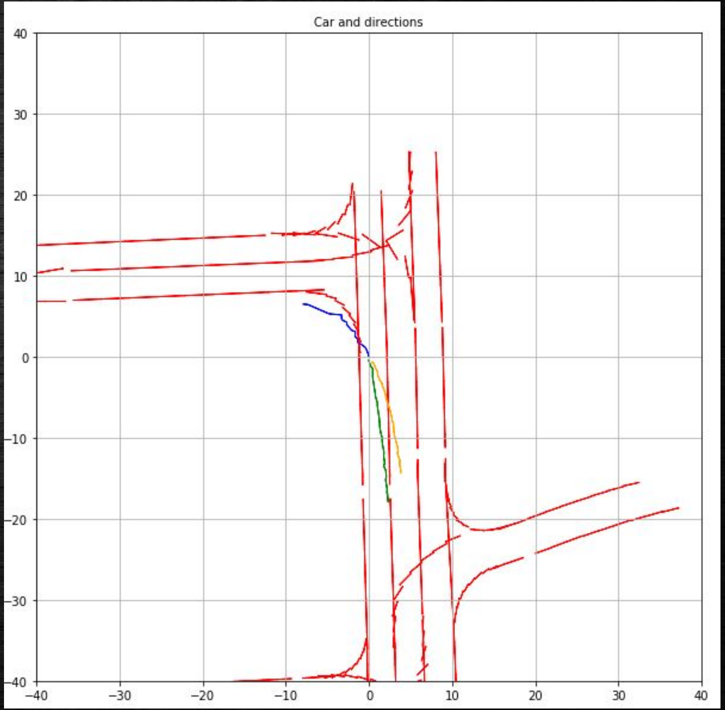
Outcome is not disrupted by the last p_in/v_in

Result of repeat-layers linear model

Training	Validation	Testing
2.18476	2.28733	1.99828



Can't handle turning very well



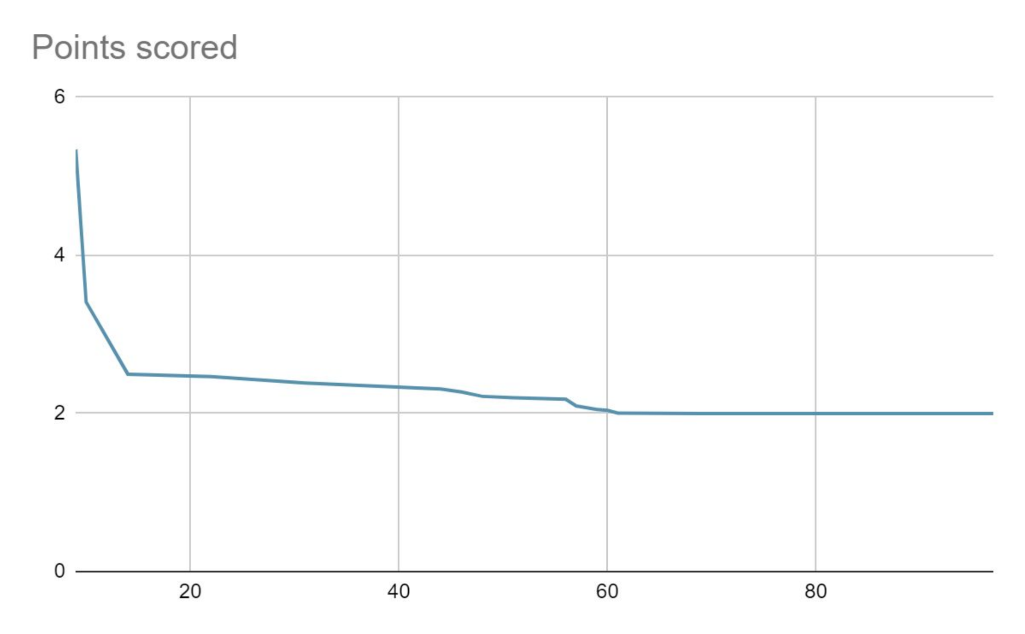
Note: blue is p_in, green is p_out, yellow is our output

Discussion

What we learned...

- Know your data before writing any code
- Experience with different models. Simpler models could work better!
- Toning is very important! Use a small dataset to tone your parameters.
- Adam is generally the best optimization algorithm.
- Compute validation score after each epoch.

Our kaggle score vs number of submissions



Future Work

- Better data pre-processing: handle the error values

- Trying more models...

- Other topics: different cities have different driving habits.

Thanks For Watching!