Motion Prediction in Autonomous Vehicles Report

Adit Sawant, Tushar Jain

{f20180234, f20190110} @pilani.bits-pilani.ac.in

November 25, 2020

1 Introduction

This paper is a summary of the various approaches we used in the Kaggle Competition: **Lyft Motion Prediction for Autonomous Vehicles** [1]
We build robust prediction models for self-driving vehicles. Our goal is to explore various methods to predict the motion of autonomous vehicles in presence of multiple agents. For this task, we need a large enough dataset with a large training dataset. Fortunately, Lyft has the largest publicly available dataset [3] for this purpose. A shortened Kaggle-friendly version of the same is available in the Data section on the main competition page. [1]

The combining process was simple when dealing with single-mode models. We consider three models, one for each channel, and tune their probabilities accordingly.

Things become harder when working with multi-mode models where there is no obvious order between different models' predictions. We try to resolve this issue by sorting the three channels such that the one with the highest probability will be the first one for all the models. Then, the channel with the second highest probability will be the second channel for all the models; and similarly for the third channel as well. In this way, we are able to combine different models.

As the whole process was slower with pure Pandas, we switch to NumPy to speed up everything.

More importantly, we stress on the recent developments in the field and review various techniques widely used for the autonomous vehicles. We particularly focus on PointNet [8] authored by Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. The paper, published in April 2017, introduces the PointNet model which has vast applications in many domains, including our problem statement.

2 Dataset

The Lyft Motion Prediction for Autonomous Vehicles competition is fairly unique, data-wise. In this competition, a very large amount of data is provided, that can be used in many different ways. Reading the data is also complex. We

refer to Lyft's L5Kit module and sample notebooks [6] to properly load the data and use it for training.

The data is packaged in .zarr files. These are loaded using the zarr Python module, and are also loaded natively by l5kit. Each .zarr file contains a set of:

- scenes: driving episodes acquired from a given vehicle.
- frames: snapshots in time of the pose of the vehicle.
- agents: a generic entity captured by the vehicle's sensors. Note that only 4 of the 17 possible agent label_probabilities are present in this dataset.
- agents_mask: a mask that (for train and validation) masks out objects that aren't useful for training. In test, the mask (provided in files as mask.npz) masks out any test object for which predictions are NOT required.
- traffic_light_faces: traffic light information.

We are predicting the motion of the objects in a given scene. For test, you will have 99 frames of objects moving around will be asked to predict their location in the next 50.

Files:

- aerial_map - an aerial map used when rasterisation is performed with mode "py_satellite"
- semantic_map - a high definition semantic map used when rasterisation is performed with mode "py_semantic"
- sample.zarr - a small sample set, designed for exploration
- train.zarr - the training set, in .zarr format
- validate.zarr - a validation set (roughly the size of train)
- test.csv - the test set, in .zarr format
- mask.npz - a boolean mask for the test set. All and only the agents included in the mask should be submitted
- *sample_submission.csv - two sample submissions, one in multi-mode format, the other in single-mode

3 Implementation

Initially, we implemented the Residual Network-based model. After that we tried to incorporate some changes in the model-related parameters so that we could get better results. We found that varying the Image and Raster size has a significant effect on the model's performance.

Observations:

On increasing the raster size, while keeping the pixel size constant, the model (agent) will "see" more of the surrounding area. (See Figure 1.) Raster size depends on the vehicle's velocity in the following manner:

km/hr	m/s	Distance in 5 sec	In pixels
1	0.28	1.39	2.78
5	1.39	6.94	13.89
10	2.78	13.89	27.78
15	4.17	20.83	41.67
20	5.56	27.78	55.56

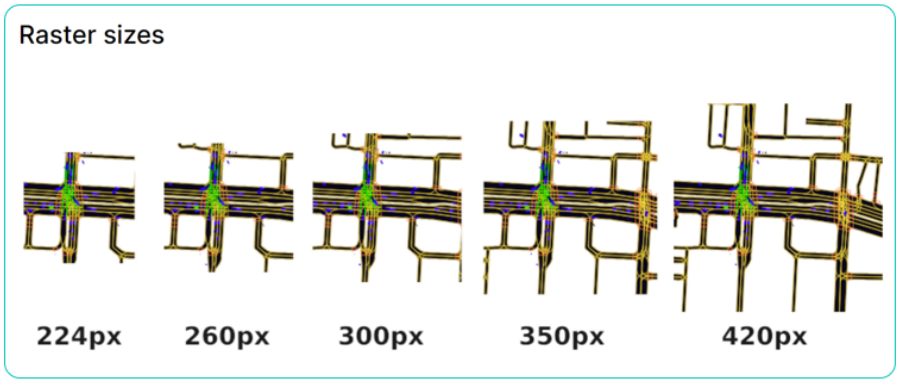

Figure 1: Varying Raster Sizes

In the Figure 2, the differences between various pixel sizes are evident. Finally, pixel_size value of somewhere between 0.1 and 0.25 worked well in this case.

Another approach:

Our use-case consisted of several birds-eye view images of the autonomous vehicles along with surrounding agents. These agents and the main AV are essentially unordered agents. They do not occur in a specific order. We can correlate this scenario with that of a point cloud.

As mentioned in the original paper - *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation*, [8] Point clouds are simple and unified structures that avoid the combinatorial irregularities and complexities of meshes, and thus are easier to learn from.

The essence of the PointNet model is its use of a single symmetric function - Max Pooling. The model selects few points based on the amount of information they add to the system. These points are encoded along with the reason for their selection. The fully connected (FC) layers in the tail of the model successfully aggregate these learnt values which can later be used for any specific use-cases.

We have used the PointNet Architecture to predict the motion of AVs based on the surrounding agents. For this problem statement, the general approach for data representation is Rasterization. The main advantage of using PointNet architecture is that it works directly on the agents dataset. Having no GPU access, we were able to achieve a decent accuracy by training the model for about 8 hours only.

The PointNet model uses a combination of feed-forward (Conv1d) models along with some transformation matrices and symmetric functions (max pool-

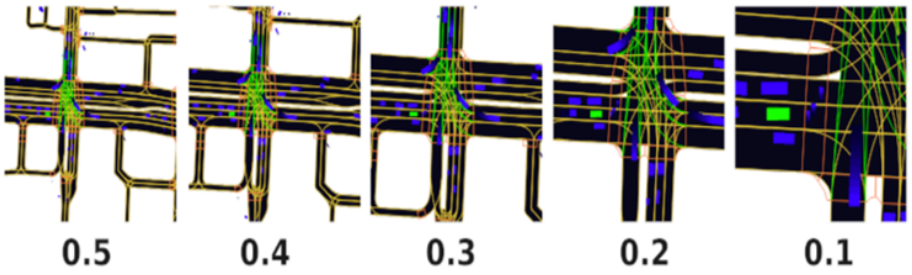

Figure 2: Varying Pixel Size

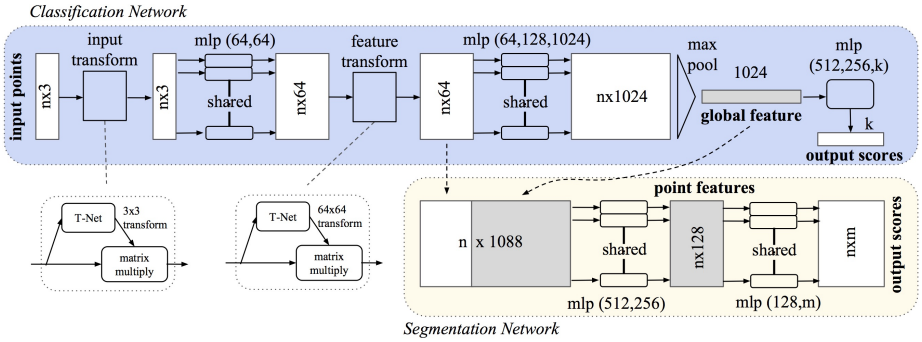

Figure 3: PointNet Architecture

ing) to deal with unordered set of points (agents). The complete architecture diagram is shown in Figure 3.

4 Results

The competition used a very unique metric for evaluation purposes. It was defined as follows -
Due to the high amount of multi-modality and ambiguity in traffic scenes, the used evaluation metric to score this competition is tailored to account for multiple predictions. We calculate the negative log-likelihood of the ground truth data given the multi-modal predictions. Let us take a closer look at this. Assume, ground truth positions of a sample trajectory are

$$x_1, \ldots, x_T, y_1, \ldots, y_T$$

and we predict K hypotheses, represented by means

$$\bar{x}_1^k, \ldots, \bar{x}_T^k, \bar{y}_1^k, \ldots, \bar{y}_T^k$$

In addition, we predict confidences c of these K hypotheses. We assume the ground truth positions to be modeled by a mixture of multi-dimensional

independent Normal distributions over time, yielding the likelihood

$$
\begin{aligned}
& p\left(x_{1,\ldots,T}, y_{1,\ldots,T} \mid c^{1,\ldots,K}, \bar{x}^{1,\ldots,K}_{1,\ldots,T}, \bar{y}^{1,\ldots,K}_{1,\ldots,T}\right) \\
&= \textstyle\sum_k c^k \mathcal{N}\left(x_{1,\ldots,T} \mid \bar{x}^k_{1,\ldots,T}, \Sigma = 1\right) \mathcal{N}\left(y_{1,\ldots,T} \mid \bar{y}^k_{1,\ldots,T}, \Sigma = 1\right) \\
&= \textstyle\sum_k c^k \prod_t \mathcal{N}\left(x_t \mid \bar{x}^k_t, \sigma = 1\right) \mathcal{N}\left(y_t \mid \bar{y}^k_t, \sigma = 1\right)
\end{aligned}
$$

which results in the loss

$$
\begin{aligned}
L &= -\log p\left(x_{1,\ldots,T}, y_{1,\ldots,T} \mid c^{1,\ldots,K}, \bar{x}^{1,\ldots,K}_{1,\ldots,T}, \bar{y}^{1,\ldots,K}_{1,\ldots,T}\right) \\
&= -\log \textstyle\sum_k e^{\log\left(c^k\right) - \frac{1}{2}\sum_t \left(\bar{x}^k_t - x_t\right)^2 + \left(\bar{y}^k_t - y_t\right)^2}
\end{aligned}
$$

The performance of various models was checked by running them on the local machine. The results have been summarised in the following table:

Model backbone	Score
Baseline	124.693
PointNet	46.804
Multi-mode	66.189
Ensemble	66.024

5 Conclusion

This problem statement is related to Autonomous "Self-driving" vehicles. It was presented in the Kaggle competition - "Lyft Motion Prediction for Autonomous Vehicles." During the course of the competition, we got to a learn a lot of new techniques of Deep Learning out there. We went on to study around 8-9 research papers related to the domain in order to get a hold of all the research done in this field. Considering the other submissions made on Kaggle, we explored various approaches and experimented with them.

Given the hardware constraints, our results are reasonable fine. We played around with the raster and image size in order to determine the optimum values of the two. However, there is still scope for improvement by training the model on the entire dataset as a whole. Based on further reading, we found that the models - Xception71, Xception65, Xception41, EfficientNetB5 seem to provide much better results. This approach can be further explored.

References

[1] Lyft motion prediction for autonomous vehicles. https://www.kaggle.com/c/lyft-motion-prediction-autonomous-vehicles.

[2] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, NITIN SINGH, and Jeff Schneider. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2020.

[3] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, A. Jain, S. Omari, V. Iglovikov, and P. Ondruska. One thousand and one hours: Self-driving motion prediction dataset. https://level5.lyft.com/dataset/, 2020.

[4] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska. One thousand and one hours: Self-driving motion prediction dataset. *arXiv preprint arXiv:2006.14480*, 2020.

[5] Siddhesh Khandelwal, William Qi, Jagjeet Singh, Andrew Hartnett, and Deva Ramanan. What-if motion prediction for autonomous driving. *arXiv preprint arXiv:2008.10587*, 2020.

[6] Lyft. L5kit. https://github.com/lyft/l5kit, 2020.

[7] S. Mandal, S. Biswas, V. E. Balas, R. N. Shaw, and A. Ghosh. Motion prediction for autonomous vehicles from lyft dataset using deep learning. In *2020 IEEE 5th International Conference on Computing Communication and Automation (ICCCA)*, pages 768–773, 2020.

[8] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[9] Neil Traft, Skanda Shridhar, Galen Clark Haynes, and ATG Uber. Motion prediction for self-driving needs a metric specific to self-driving.

[10] Linjun Zhang and Eric Tseng. Motion prediction of human-driven vehicles in mixed traffic with connected autonomous vehicles. In *2020 American Control Conference (ACC)*, pages 398–403. IEEE, 2020.