Temwirk : Motion Prediction for Autonomous Vehicles

Vignesh K Kumar
PES120180085
PES University
Department of CSE
vigneshkkumar127@gmail.com

Chandratop Chakraborty
PES120180062
PES University
Department of CSE
chandratop.mail@gmail.com

Tarun Gupta J
PES120180073
PES University
Department of CSE
tarunguptajsts@gmail.com

Arun Srinivasan P
PES1201800383
PES University
Department of CSE
arunsrinipartha@gmail.com

***Abstract*—Autonomous vehicles have attracted remarkable attention from both public perceptions as well as an industrial standpoint. While the idea of autonomous vehicles has been around for a long period of time it is only now that it has started to become truly viable due to the explosion in data with the growth of the internet and smart devices. Designing an autonomous vehicle requires a highly accurate motion prediction system that considers all possible scenarios. An important task is to understand the 3D properties of the vehicle such as translation, rotation, and shape in order to successfully predict the position of the vehicle. However to attain a truly autonomous system we would be required to understand the agent-environment interaction and consider the inherent uncertainty associated with driving. This understanding could be obtained by training a deep network with a vast amount of well structured data related to the agents and the scenes. To accomplish this task we are working on the Lyft Driving Dataset.This dataset is the largest collection of the traffic agent motion data. It includes the logs of movement of cars, cyclists, pedestrians, and other traffic agents encountered by Lyft's autonomous fleet. These logs come from processing raw lidar, camera, and radar data through our team's perception systems and are ideal for training motion prediction models. Finally, the task is to predict the motion of external objects such as cars, cyclist, pedestrians etc in order to assist the self-driving car.**

***Index Terms*—Autonomous Vehicles, Rasterization, Scene Understanding, Motion Prediction**

I. Introduction

With the rise of the digital age and the increasing interconnectivity across all spheres of life, humankind has seen exponential progress in every facet. However, despite this progress, driving remains an unpredictable and potentially fatal task. The key cause for this risk is human error and a lack of attention to one's surroundings as well as an inability to process all the relevant information while driving. This human error however is to a certain extent unavoidable, the only way to truly minimize this issue is to utilize systems that accurately predict the positions of all actors involved in a certain scene and chooses the appropriate action at all times. In this way, Autonomous Vehicles (AVs) are expected to dramatically redefine the future of transportation. However, there are still significant engineering challenges to be solved before one can fully realize the benefits of self-driving cars. One such challenge is building models that reliably predict the movement of traffic agents around the AV, such as cars, cyclists, and pedestrians.

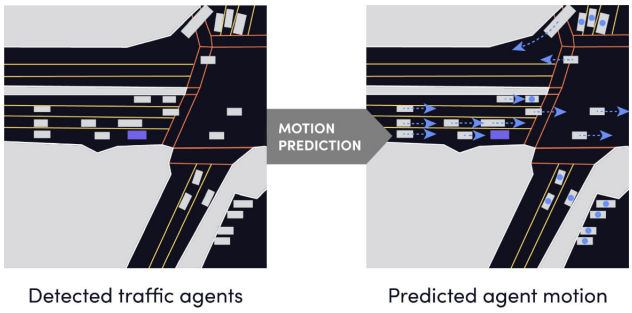

Fig: The above image visualizes motion prediction that is accomplished for agents in the traffic scenes from available data.

Today, models for motion prediction and planning are mainly built using rule-based systems. However, the future is uncertain and rules may not always scale well with uncertainty. By adding other agents into the mix, the number of rules and costs grow exponentially. In theory, an automated vehicle system can only be termed as an "autonomous" system, when all the dynamic driving tasks, at all driving environments, can be performed by the vehicle's automated system. This of course requires an obscene amount of data that has to be well structured and must span practically every environment possible.

The potential impact of autonomous vehicles on human life cannot be understated. AVs will completely restructure future

travel, and the design of roads, parking facilities and public transit services. Autonomous vehicles must be sufficiently reliable, affordable and common to displace most human driving. It is essential that the technology provide huge savings and benefits for it to be widely appropriated. There is considerable uncertainty concerning autonomous vehicle development, benefits and costs, travel impacts, and consumer demand. Considerable progress is needed before autonomous vehicles can operate reliably in mixed urban traffic, heavy rain and snow, unpaved and unmapped roads, and where wireless access is unreliable. Years of testing and regulatory approval will be required before they are commercially available in most jurisdictions. The first commercially available autonomous vehicles are likely to be expensive and limited in performance. They will introduce new costs and risks. These constraints will limit sales. Many motorists will be reluctant to pay thousands of extra dollars for vehicles that will sometimes be unable to reach a destination due to inclement weather or unmapped roads.

II. Related Work

In this section we review past works performed to accomplish the task at hand and we summarize the aspects useful for our work.

Houston et al. [1] worked on the Self-driving Motion Prediction Dataset. In this paper they outline the features and key aspects of the dataset and explain in-depth the methods used to compile the dataset. The dataset consists of scenes, where each scene is 25 seconds long and captures the perception output of the self-driving system, which maps the precise positions and motions of nearby vehicles, cyclists, and pedestrians over time. The dataset is encoded in the form of n-dimensional compressed Zarr arrays, which enables fast random access throughout the dataset. The data is collected using a fleet of self-driving vehicles driving along a fixed route. The sensors utilized for the perception include seven cameras, three LiDARs, and five radars. Apart from the scenes, the dataset also comprises of Aerial maps and semantic maps. The aerial maps capture the area that runs along the route at a resolution of 6 cm per pixel[3]. The semantic maps capture information about the roads and traffic elements. The entire dataset is structured in a manner apt for motion prediction tasks. Additionally, a python toolkit named L5kit is released, alongside for accessing the dataset. It provides useful features such as Multi-threaded data loading and sampling, customisable scene visualization and rasterization, and Baseline motion prediction solution.

Hong et.al. [2] worked on the prediction of driving behaviour with a convolutional model of semantic interactions. This paper discusses the problem of predicting future states of entities in complex, real-world driving scenarios. Many of the previous research papers have addressed mainly on how they predict small future time intervals. These papers also take input as raw sensory information(camera, LiDAR, or radar) which requires a heavy emphasis on extracting high-level representation of entities.Moreover, the publicly available datasets used are very small and unrealistic. The dataset provided by this paper includes the following- 9,659 unique vehicles in 83,880 prediction scenarios (173 hours), in 88 physically-distinct locations and includes semantic map information. Using the given road data which includes connectivity of roads, lanes, junctions, stop and yield lines, etc., an RGB image that contains elements with unique colors is mapped to its geometric primitives. A neural network is used to map the low-level sensor information to 3D tracked entities. The output model is represented by a probability distribution over the entity state space at each time step, the actions the entity might take at a particular time, and efficiently predicting the full trajectories of the entity. The industry and linear baselines performed worse compared to these methods for predicting a large time interval(5 seconds into the future), but better in smaller intervals. A useful insight from the paper is a dataset with a wide variety of real-world locations and unique 3D tracks are necessary for future predictions. Also, the representation of multimodality is crucial in determining real-world planning for driving to avoid collisions and hence accidents.

Djuric et al. [3] worked on a motion prediction model while considering the inherent uncertainty encountered while making predictions in this space. This paper provides an approach to solve one of the integral aspects of the deployment of self-driving vehicles, which is the prediction of the actions taken by the various actors (vehicles, pedestrians, etc) on the scene. The approach used consists of two major phases, the rasterization of the maps and surroundings in the vehicles' vicinity followed by the training of a deep convolutional neural network for the prediction of the short-term trajectory for the actors while accounting for the inherent uncertainty in the environment. The rasterization process allows the complex 3D scene to be modeled in a more understandable manner, this is accomplished by utilizing a vector layer to represent the different major parts of the scene such as the roads or vehicles, and then assigning a distinct RGB colour to each of these layers. This rasterization process returns a 2D aerial view with distinct colours from the complex 3D scene which allows it to be more easily parsed by the neural network. The network architecture extracts features from the rasterized results and then passes it through two fully connected layers to obtain the trajectory prediction.

Ivanovic et al. [4] provided a paper with some important insights into how to handle driving data, namely with the rasterization procedure as well as the inherent uncertainty in the environment which must be taken into account while making predictions.The goal of conditional generative modeling is to fit a model of the conditional probability distribution p(y — x), which may be used for downstream applications such as inference, or to generate new samples y given x. The encoder neural network, parameterized by i, takes the input x and produces a distribution pi(z — x)

where z is a latent variable that can be continuous or discrete. The decoder neural network, using the same parameter, uses the input x and samples from the encoder to produce to produce pi(y — x, z) conditional probability. p(y — x) is then obtained by marginalization of the latent variable z. RNNs are leveraged to process time series data without increasing problem size.Future work on this paper includes developing ways to make the latent space more interpretable using better temporal logic, make the model fit against upstream sensor noise.

III. Problem Statement

The task at hand is to predict the motion of external objects such as cars, cyclist, pedestrians etc in order to assist the self-driving car. We have to predict the location of agents in the next 'n' frames. Agents can be other cars, cyclists, pedestrians as well as any other denizen of the road. Each of these agents are associated with specific movements and motion patterns. For instance a car would typically move at a much higher speed when compared to a pedestrian, however the turning radius of a car will also be proportionally higher. All these parameters must be considered to make accurate and viable predictions. It is not enough to make a singular prediction as each state of the environment could lead to a wide variety of possible future states. The dataset we will utilize for this procedure is the Lyft driving dataset which provides us with enough structured information to make meaningful predictions.

A. Dataset

The Lyft Driving dataset consists of four major structures i.e. scenes, frames, agents and traffic light faces which are stored in zarr files. In Zarr datasets, the arrays are divided into chunks and compressed. We have performed a basic EDA on a sample of these structures to understand internal relations. The dataset in an overall sense has no missing data and as a whole contains no inconsistent data.

1) Agents: An agent is an observation by the aerial view of some other detected object. Each entry describes the object in terms of its attributes such as position and velocity while also giving the agent a tracking number to track it over multiple frames as well as its most probable label. While there are 17 possible labels numerically only 3 have a name (cyclist, pedestrians and cars) and the others are grouped as unknowns. The schema is as follows:

- Centroid - position of agent (Given as x and y separately)
- extent - agent dimension (Given as x and y separately)
- yaw - rotation of an agent with respect to vertical axis. A yaw rotation is a movement around the yaw axis of a rigid body that changes the direction it is pointing, to the left or right of its direction of motion.
- velocity - speed of the agent (Given as x and y separately)
- track_id - unique id to track agent in different frames
- label_probabilities - probability an agent belongs to one of the 17 classes. (We are only given three labels)

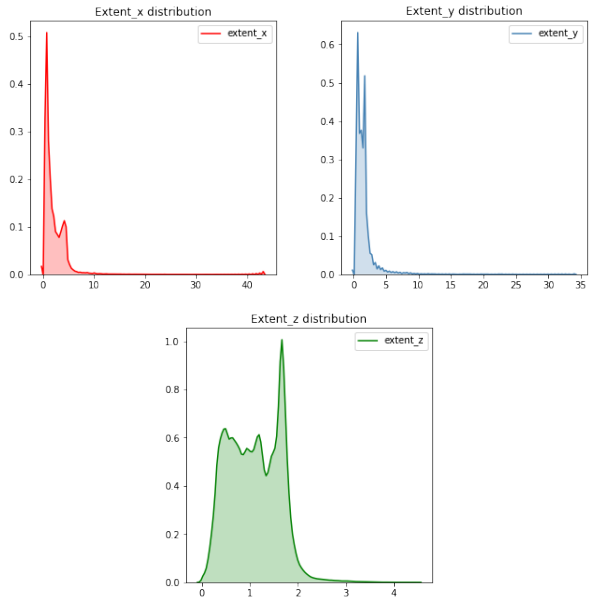

Fig: Distribution of the extents in x, y and z. It has long tails in the positive direction hinting at a right skewed distribution. Whereas the extent of skewness varies among the 3 coordinates.

2) Scenes: A scene is identified by the host (i.e. which car was used to collect it) and a start and end time. It consists of multiple frames (=snapshots at discretized time intervals). The scene datatype stores references to its corresponding frames in terms of the start and end index within the frames array (described in dataframe below). The frames in between these indices all correspond to the scene (including start index, excluding end index).The schema is as follows:

- frame_index_interval - frame index (including start index, excluding end index)
- host - car used to collect data
- start_time - start time of scene
- end_time - end time of scene

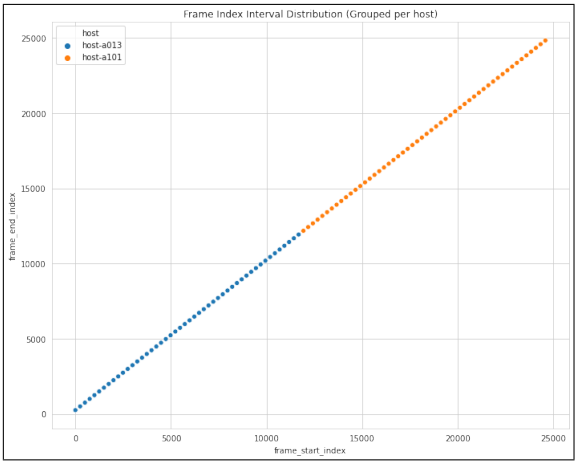

Fig: We can see two host cars namely "host-a013" and "host-a101" were utilized to collect the samples for the scenes. We can also see a linear relation between the two attributes.

3) Frames: A frame captures all information that was observed at a given instance of time. The schema is as follows:

- timestamp - frame's timestamp
- agent_index_interval - agents (vehicles, cyclists and pedestrians) that were captured by the ego's sensors
- traffic_light_faces_index_interval - traffic light index
- ego_translation - position of host car.
- ego_rotation - rotation of host car (which is collecting data using ego sensors)

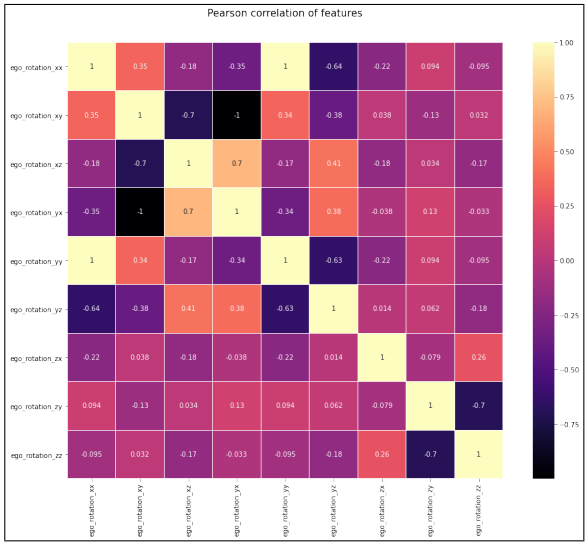

Fig: The above is a visualization of the correlation between the 'ego-translation' attributes. This is in other words a heatmap to give us a visual representation of the confusion matrix.

4) Traffic: The traffic light bulbs (red, green, yellow) are referred to as faces. The schema is as follows:

- face_id - unique id for traffic light bulb
- traffic_light_id - traffic light status
- traffic_light_face_status - out of red/green/yellow which face is active/inactive/unknown

IV. Proposed Methodology

The Lyft dataset consists of detailed information about the environment and agents in the form of zarr files. This information has been obtained from special aerial maps as well as semantic maps of the locations which have been processed and converted into easy to parse numerical data. This allows us to make informed predictions solely based on the scene data without resorting to complex image processing algorithms to extract this data. The aerial maps consist of extremely high resolution images with dimensions in the range of 12000x9000. These high resolution images can be extensively zoomed into to extract a vast amount of information about the position of the agents as well as the environment. This extraction process is a crucial part of the pre-processing stage, and it is only after this information is organized that we can begin training the model with the relevant data.

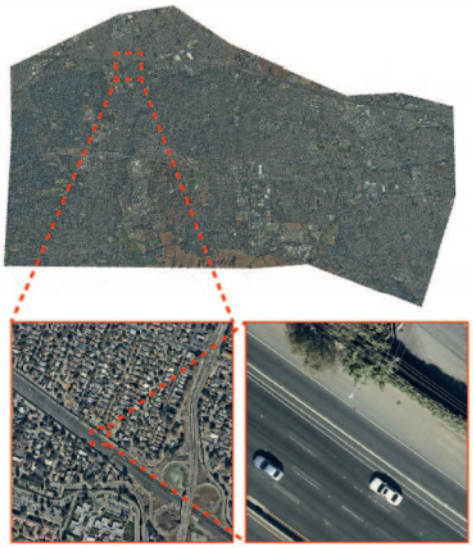

Fig : Representation of the aerial map and the level of zoom needed to extract the relevant information.The image on top is the entire aerial map, the image present at the bottom left is an example of near map image (smaller segment), the image on the bottom right is the level of zoom possible for us to extract useful information. Image taken from [1].

We utilized the data stored in the zarr files, that is the data related to the scenes, to make our predictions. The key aspects which are relevant for our predictions are the centroid as well as the extent of the agents as this tells us the total space occupied by the agent as well as the position. The velocity of the agent will allow us to make predictions on the path it is currently on. This information is used in tandem with the information in the frames file which tells us the timestamp and allows us to link the information between all 4 files. The traffic lights along with the timestamp help us predict if an agent is going to stop or start moving as well. The data is parsed using the L5toolkit provided which allows us to easily understand the data and make predictions.
We utilized the ResNet [5] backbone to make the predictions of the path taken. Our model returns three possible trajectories alongside the confidence score for each trajectory.

A. ResNet

ResNet allows us to overcome the vanishing gradient problem by utilizing an "identity shortcut connection" [5] ie it skips one or more layers. These shortcut layers allow us to avoid the problem where the gradients become NaN in our computations and allow us to design sufficiently deep networks to make accurate predictions. We need to use the loss function that takes that into account the degree of variance of each, simply using RMSE does not lead to an accurate model. We have utilized negative log likelihood for this purpose.

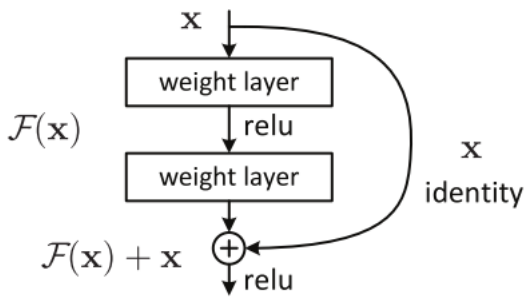

Fig: The above is a visualization of the residual block used by ResNet.

B. Approach

The main problem that we are trying to solve involves predicting the coordinates of the centroids for all agents for a given timestamp. Due to the inherent uncertainty in the problem we have made three distinct predictions for each agent alongside a confidence value for each prediction. The processing of the data is mainly accomplished using the L5Toolkit which allows us to accurately model and understand the scenes.

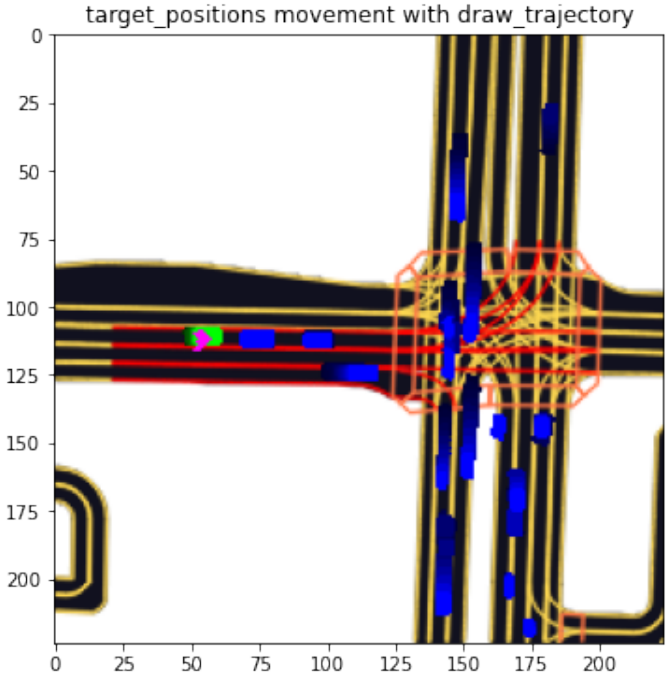

Fig: The above is a visualization of the tracks after rasterization.

Our main pre-processing involves the rasterization of the scenes, which allows the problem to be visualised in a much clearer manner as shown above. The main predictions however are made from the numerical data, which consists of in-depth information about all agents in the scene. Our model takes in this frame data as input and also maintains a specified number of previous frames as history to aid in making predictions. These history frames are integral in making accurate predictions and are the key factor which allow us to predict multiple trajectories. These history frames help us identify which path was predicted and allows the different trajectories to be consistent. After the output frames are generated for each trajectory they are compared with the actual output to compute the confidence score for each trajectory.

We experimented with various different structures and have included multiple variants of the ResNet model such as ResNet18, ResNet34 and ResNet50.

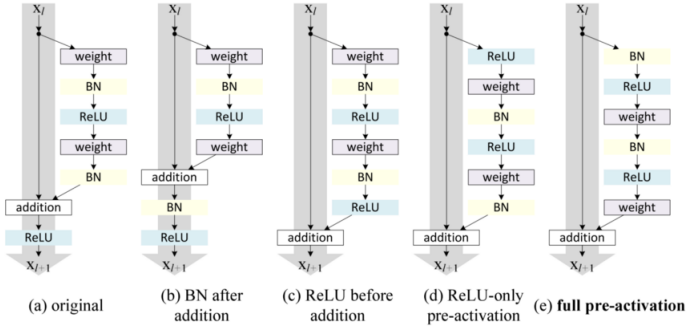

Fig: The above is a visualization of different types of residual blocks.

V. Experimental Results

Due to the high degree of ambiguity involved in the prediction of traffic scenes, the evaluation metric accounts for the prediction of multiple different trajectories. First the ground truth positions of a sample trajectory are taken as follows:

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

Then the predicted hypotheses are represented as means:

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

The likelihood of these predictions are modelled by assuming the ground truth positions to be a mixture of multi-dimensional independent normal distributions over time. This likelihood and loss is as shown below.

$$p(x_{1,\ldots,T}, y_{1,\ldots,T} | c^{1,\ldots,K}, \bar{x}_{1,\ldots,T}^{1,\ldots,K}, \bar{y}_{1,\ldots,T}^{1,\ldots,K})$$

$$= \sum_k c^k \mathcal{N}(x_{1,\ldots,T} | \bar{x}_{1,\ldots,T}^k, \Sigma = 1) \mathcal{N}(y_{1,\ldots,T} | \bar{y}_{1,\ldots,T}^k, \Sigma = 1)$$

$$= \sum_k c^k \prod_t \mathcal{N}(x_t | \bar{x}_t^k, \sigma = 1) \mathcal{N}(y_t | \bar{y}_t^k, \sigma = 1)$$

$$L = -\log p(x_{1,\ldots,T}, y_{1,\ldots,T} | c^{1,\ldots,K}, \bar{x}_{1,\ldots,T}^{1,\ldots,K}, \bar{y}_{1,\ldots,T}^{1,\ldots,K})$$

$$= -\log \sum_k e^{\log(c^k) + \sum_t \log \mathcal{N}(x_t | \bar{x}_t^k, \sigma=1) \mathcal{N}(y_t | \bar{y}_t^k, \sigma=1)}$$

$$= -\log \sum_k e^{\log(c^k) - \frac{1}{2} \sum_t (\bar{x}_t^k - x_t)^2 + (\bar{y}_t^k - y_t)^2}$$

This evaluation parameter is provided in the L5Toolkit and gives a final number as the score. We obtained a score of 23.610. This score isn't a direct representation of accuracy as the problem statement contains a considerable amount of inherent ambiguity. A lower score signifies better predictions, for comparison, the best score was around 9.310. Our ranking is in the initial 300's in the leader-board.

Our model does relatively well, considering the total amount of participants was around 950. The main areas where our model works well are in situations where there aren't too many potential trajectories and the path to be predicted is relatively straightforward, for example, a vehicle traveling on a straight highway. However, due to the increased amount of noise and the sheer degree of variance in other situations, such as at a four-way intersection, our model was not as successful as it was in straightforward situations.

Other approaches that we experimented with include using PCA to reduce the dimensionality to a more manageable level before making predictions, this however did not lead to an improvement and hence we decided not to go forward with it. We also tried to add the velocity and yaw as parameter to the final layer of the network, to provide a higher to those parameters, however this was also not viable. We read up on other networks that could have been potentially provided better results such as mobilenetv3 and densenet121 but we did not implement these models. We also experimented with different raster sizes but we settled on the raster size of 224,224 for our results.

VI. Conclusions

The entire project was done using online collaborative platforms such as overleaf, discord and kaggle notebooks. As the dataset was excessively large we did our processing exclusively on cloud services and not on our local systems. Regarding each team members contributions, we joined a voice call with the screen shared to simulate working together in the same environment. This allowed us to freely exchange ideas and brainstorm. Therefore the total contribution from each member is equal as no specific content can be attributed to a single member. One exception is that we each researched and summarized a different paper as mentioned in the related work section.

References

[1] 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," *arXiv preprint arXiv:2006.14480*, 2020.

[2] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," 2019.

[3] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," in *The IEEE Winter Conference on Applications of Computer Vision*, pp. 2095–2104, 2020.

[4] B. Ivanovic, K. Leung, E. Schmerling, and M. Pavone, "Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach," *arXiv preprint arXiv:2008.03880*, 2020.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.