Motion Prediction for Autonomous driving

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY
(BANARAS HINDU UNIVERSITY)
VARANASI

AUTHORS

Aloori Raj Aryan (18085005)

Bodaballa Vamsi Krishna (18085018)

SUPERVISOR

Dr. Avirup Maulik

Assistant Professor

Department of Electrical Engineering

Abstract

To be able to understand the dynamic driving environment, an autonomous vehicle needs to predict the motion of other traffic participants in the driving scene. Motion prediction can be done based on experience and a recently observed series of past events and entails reasoning about probable outcomes with these past observations. Aspects that influence driving behavior comprise many factors, such as general driving physics, infrastructure geometry, traffic rules, weather, and so on.
Models that are used today are incapable of including many of these aspects. These deep learning models often rely heavily on the past-driven trajectory as an information source for future prediction. Sparsely, some work has been done to include interaction between vehicles or some infrastructural features to the model. The lack of information regarding the driving scene can be seen as a missed opportunity because it is used-full feature source to predict the motion of a vehicle. Especially when a map of the driving scene is available, reliable information regarding the road layout can be extracted.
In this thesis, a model architecture is sought that is aware of the interaction between vehicles, and that understands the geometry of the roads in the scene. Importantly, map information regarding the driving scene should be used as the primary source of information regarding the road geometry. First, a baseline deep learning model is constructed, that can generate predictions based on the past observed trajectory. To add interactive features to the model, a social pooling module is introduced. The social pooling method allows to efficiently include the driving behavior of other vehicles in the scene.
To introduce reliable, map-based road information to the model, two novel methods are proposed. In the first, the predictions from a model are used to extract features in a map. These features describe the road scene around the predicted location and are used to update these predictions. In the second method, the semantic map is only used to extract a road segment ahead of the vehicle. A road-RNN is introduced to construct features regarding the road segment, and an attention mechanism to determine what part of the road segment is relevant for the predictions. These modules are referred to as road refinement and road attention respectively.
A baseline deep learning model is used and extended with a road-geometry module, an interaction module, a combination of the two, or none. To test the prediction capabilities of the models, they are trained on a dataset consisting of trajectory recordings from a straight highway with dense traffic.

1. Introduction

Over the past several years, significant advancements have been made in autonomous driving. One active area of research is in methods for predicting the future motion of surrounding actors in the scene. In this work, we focus on the prediction problem, and in particular, we want to direct the attention of the research community to a less explored approach: combining robotics with machine learning. We argue that this hybrid approach is useful in order to maintain a safety-critical system over a long period of time

Fully autonomous driving can only be achieved if the autonomous vehicle has a degree of understanding about the surrounding environment. The autonomous vehicle that perceives the environment is often referred to as the ego vehicle, being the center of the perceived traffic situation around it. To obtain a reliable perception of the environment, the ego vehicle relies on a number of different sensors, generally consisting of laser scanners (LIDAR), 3D cameras, regular cameras, radars, and ultrasonic sonars. These sensors enable perception of the environment by combining complementary information, such as scene vision and radar distances, which is crucial for localization, object detection, and mapping. Moreover, some autonomous vehicles such as the WEpods have a map available, with information regarding the static driving scene that is gathered offline. By means of integration of sensory and map information, an understanding of the traffic situation can be formed.

Understanding the dynamic environment is a complex challenge, and is being actively researched. To be able to understand the dynamic environment, an estimation of the driving behavior of other traffic participants has to be obtained. Estimating driving behavior can be achieved by anticipating the intentions that surrounding traffic participants have, and the motion that follows from this intention. A motion prediction module computes the expected trajectory for each traffic participant over a desired time in the future, called the prediction horizon. With the predicted trajectories, an assessment can be made whether the planned path conflicts with other trajectories or otherwise unwanted situations will arise.

Motion prediction can be done based on experience and a recently observed series of past events and entails reasoning about probable outcomes with these past observations. Aspects that influence driving behavior comprise many factors, such as general driving physics, infrastructure geometry, traffic rules, weather, and so on. This also includes the dynamic interaction between traffic participants, which is particularly important in crowded areas. In an ideal situation, all of these factors are taken into account when predicting the trajectories of vehicles in the scene.

Among all the steps motion prediction has got a unique complexity due to additional interaction with the environment. Prior to the discussion about this topic, it would be useful to clarify the technical jargon which will be used in the following sections. After the localization and perception process for the self-driving car (SDC), SDC is required to consider possible future scenarios based on its current state. All the future possible scenarios rely on the behavior of the surrounding environment and the new information which will be provided by the sensors as a planning step. Motion prediction plays a significant role in the autonomous driving application as it is a support for decision making and enables us to assess the risks associated with the planning process. In the literature, the words behavior, motion, and maneuver are used interchangeably. In the following section, the challenges with motion prediction will be discussed. After that, the terminology will be defined and finally, the probabilistic representation of the problem will be identified.

Technical Concepts and Terminology Definitions

To define the problem of the self-driving vehicle and smart vehicle, it is useful to define the technical terminologies in this area

- **Semantic Map** Every type of modeling for self-driving cars requires high-resolution maps. The semantic maps are crucial for the autonomous vehicle (AV) from an operational and safety point of view. The semantic maps are providing a better understanding of AVs from their surrounding environment. The semantic maps consist of several layers such as road networks, building in 3D, lane layer, pathways, etc.
- **Sensors** There are three major sensors installed on self-driving cars, cameras, Radar (Radio Detection and Ranging), and LiDAR (Light Detection and Ranging). The cameras are available to display highly detailed and realistic images from the surrounding environment. This would enable self-driving cars to detect objects and classify them. The radar sensor is working based on the doppler effect. There exist two types of radar sensors, long- and short-range. The long-range radar sensors (77 GHz) are controlled by automatic distance control and break distance control. The short-range (24 GHz) enables blind-spot monitoring. By LiDAR it is possible to localize self-driving cars by means of point cloud matching. The LiDAR sensor could provide a fully 360-degree map around the vehicle. This sensor detects the data from the current environment and matches it with pre-existing data in a high-definition map. This approach will yield the global position and heading the car on a semantic map. There are different algorithms to match the point cloud with a high definition map such as an interactive closest point (ICP), Filter approach, Histogram Filter, Kalman Filter.

- **Target Vehicles (TVs)** are those vehicles whose behavior is important to be predicted by the model and the self-drive car is in interaction with them.
- **Ego Vehicle (EV)** is a self-drive car that is moving while observing the behavior of the TVs
- **Surrounding Vehicles (SVs)** are whose behavior impacts the self-driving car. Choosing the SVs is adapted with different algorithms and depends on the model assumptions of Non-Effective Vehicles (NVs).

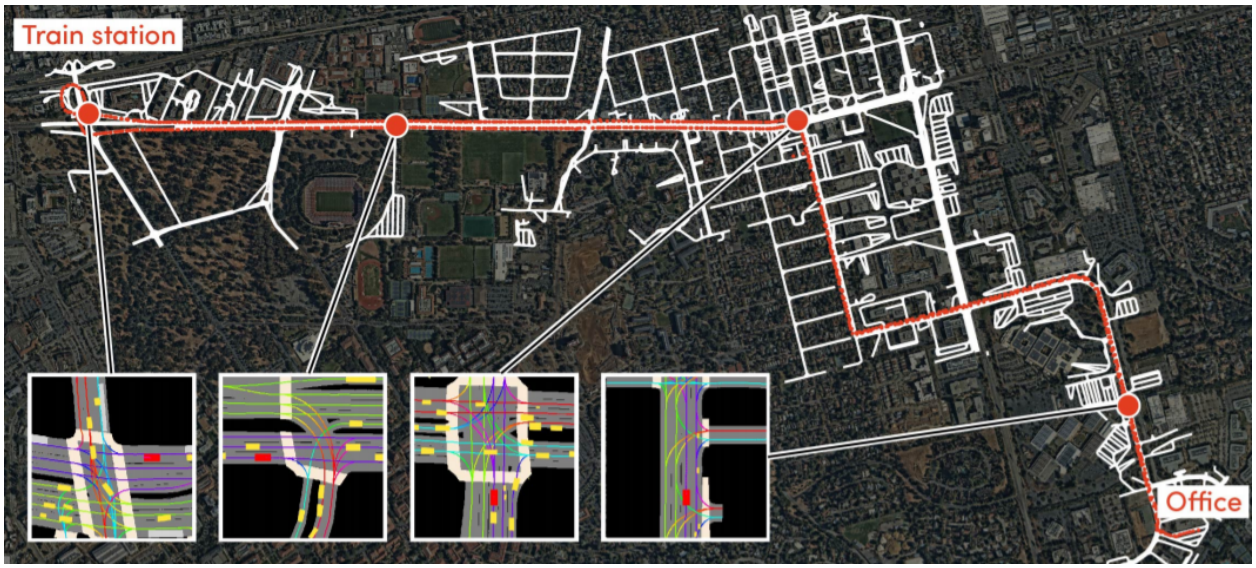
2. Dataset

The availability of large-scale datasets has been a large contributor to AI progress in the recent decade. In the field of self-driving vehicles (SDVs), several datasets, such as The KITTI dataset, Argoverse: 3d tracking and forecasting, and the Lyft level 5 AV dataset, enabled great progress within the development of perception systems. These allow an SDV to process LiDAR and camera sensors for understanding the positions of other traffic participants including cars, pedestrians, and cyclists around the vehicle. Perception, however, is only the first step in the modern self-driving pipeline. Much work remains to be done around data-driven motion prediction of traffic participants, trajectory planning, and simulation before SDVs can become a reality. Datasets for developing these methods differ from those used for perception in that they require large amounts of behavioral observations and interactions. These are obtained by combining the output of perception systems with an understanding of the environment in the form of a semantic map that contains priors over expected behavior. The broad availability of datasets for these downstream tasks is much more limited though, and mostly available only to large-scale industrial efforts in the form of in-house collected data. This limits progress within the computer vision and robotics communities to advance modern machine learning systems for these important tasks.
In this work, we share the largest and most detailed dataset to date for training motion forecasting and planning solutions. We are motivated by the scenario of a self-driving fleet serving a single, high-demand route - rather than serving a broad area. We consider this to be a more feasible deployment strategy for ridesharing since SDVs can be allocated to particular routes while human drivers serve the remaining traffic. This focus allows setting better bounds on required system performance and accident likelihood, both key factors for real-world self-driving deployment.

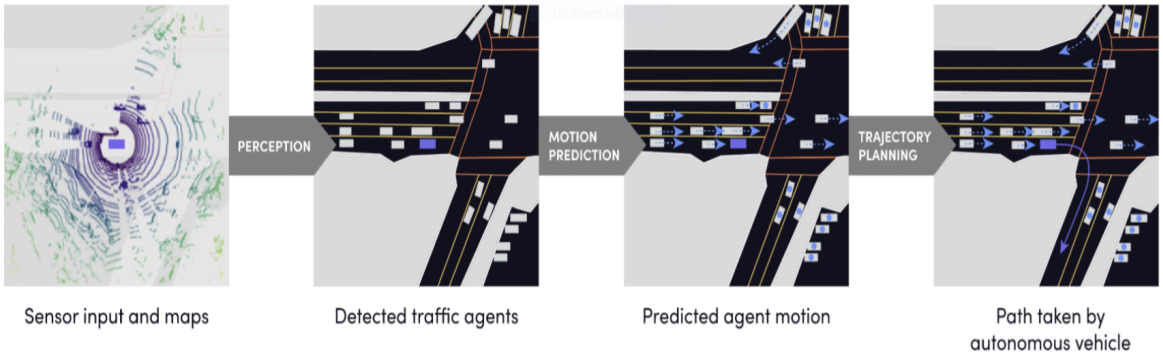
In summary, the released dataset consists of:

- The largest dataset to date for motion prediction, containing 1,000 hours of traffic scenes that capture the motions of traffic participants around 20 self-driving vehicles, driving over 26,000 km along a suburban route.
- The most detailed high-definition (HD) semantic map of the area, counting over 15,000 human annotations including 8,500 lane segments.
- A high-resolution aerial image of the area, spanning 74 km2 at a resolution of 6 cm per pixel, providing further spatial context about the environment.
- A Python software library L5Kit for accessing and visualizing the dataset.
- Baseline machine learning solutions for the motion forecasting and motion planning task that demonstrate the impact of large-scale datasets.

An overview of the released dataset for motion modeling, consisting of 1,118 hours of recorded self-driving perception data on a route spanning 6.8 miles between the train station and the office (red). The examples on the bottom left show released scenes on top of the high-definition semantic map that captures road geometries and the aerial view of the area.



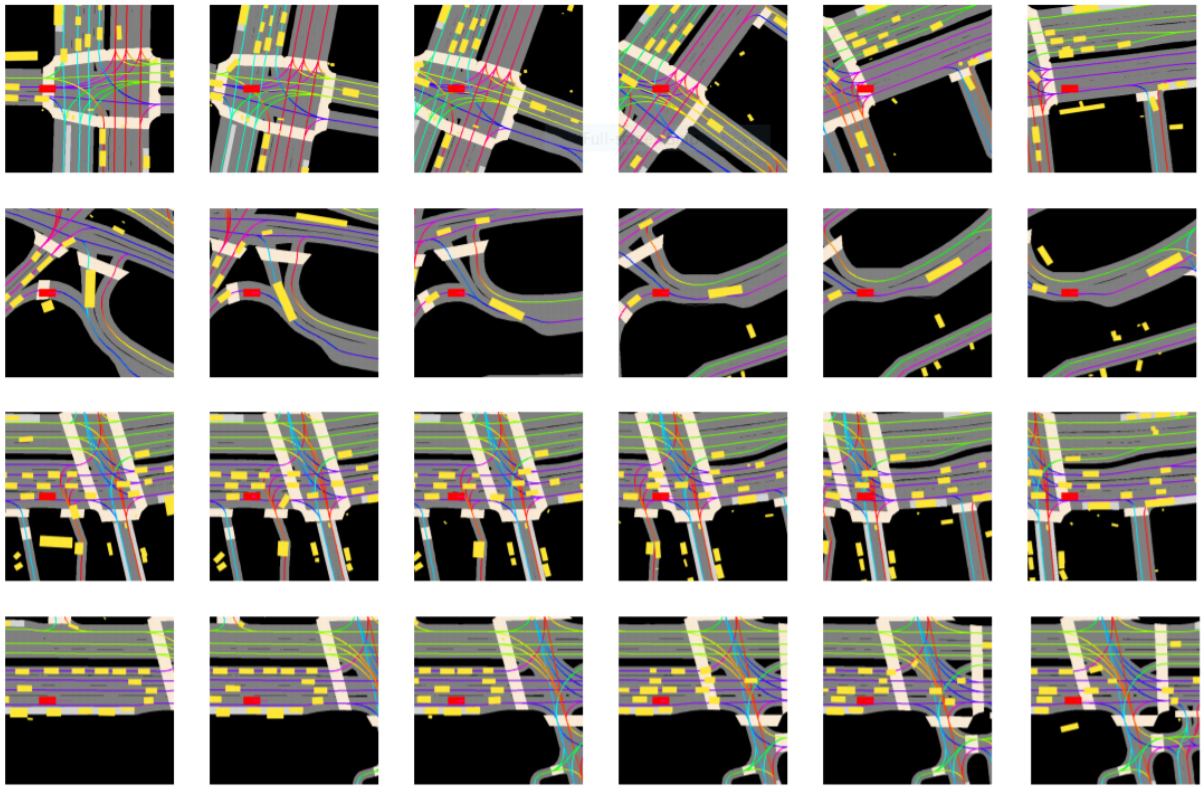
In the below figure, the raw sensor input is first processed by a perception system to estimate the positions of nearby vehicles, pedestrians, cyclists, and other traffic participants. Next, the future motion and intent of these actors are estimated (also called motion prediction or forecasting) and used for planning SDV trajectory.



Here we outline the details of the released dataset, including the process that was used to construct it. The dataset has three components:

1. 170,000 scenes, each 25 seconds long, capturing the movement of the self-driving vehicle, traffic participants around it, and traffic lights state.
2. A high-definition semantic map capturing the road rules, lane geometry, and other traffic elements.
3. A high-resolution aerial picture of the area can be used to further aid in prediction.

The dataset consists of 170,000 scenes, each 25 seconds long, totaling over 1,118 hours of logs. Example scenes are shown in Figure 4. All logs were collected by a fleet of self-driving vehicles driving along a fixed route. The sensors for perception include 7 cameras, 3 LiDARs, and 5 radars. The sensors are positioned as follows: one LiDAR is on the roof of the vehicle, and two LiDARs are on the front bumper. The roof LiDAR has 64 channels and spins at 10 Hz, while the bumper LiDARs have 40 channels. All seven cameras are mounted on the roof and together have a 360◦ horizontal field of view. Four radars are also mounted on the roof, and one radar is placed on the forward-facing front bumper.



3. Model

A. Residual Network

There have been a series of breakthroughs in the field of Deep Learning and Computer Vision. Especially with the introduction of very deep Convolutional neural networks, these models helped achieve state-of-the-art results on problems such as image recognition and image classification. So, over the years, the deep learning architectures became deeper and deeper (adding more layers) to solve more and more complex tasks which also helped in improving the performance of classification and recognition tasks and also making them robust. But when we go on adding more layers to the neural network, it becomes very much difficult to train and the accuracy of the model starts saturating and then degrades also. Here comes the ResNet to rescue us from that scenario, and helps to resolve this problem.

Residual Network (ResNet) is one of the famous deep learning models that was introduced by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang in their paper. The paper was named **"Deep Residual Learning for Image Recognition"** in 2015. The ResNet model is one of the popular and most successful deep learning models so far.

The problem of training very deep networks has been relieved with the introduction of these Residual blocks and the ResNet model is made up of these blocks. The problem of training very deep networks has been relieved with the introduction of these Residual blocks and the ResNet model is made up of these blocks. In the above figure, the very first thing we can notice is that there is a direct connection that skips some layers of the model. This connection is called 'skip connection' and is the heart of residual blocks. The output is not the same due to this skip connection. Without the skip connection, input 'X gets multiplied by the weights of the layer followed by adding a bias term. Then comes the activation function, f() and we get the output as H(x).

$$H(x)=f(wx + b) \text{ or } H(x)=f(x)$$

Now with the introduction of a new skip connection technique, the output is H(x) is changed to

$$H(x)=f(x)+x$$

But the dimension of the input may be varying from that of the output which might happen with a convolutional layer or pooling layers. Hence, this problem can be handled with these two approaches:

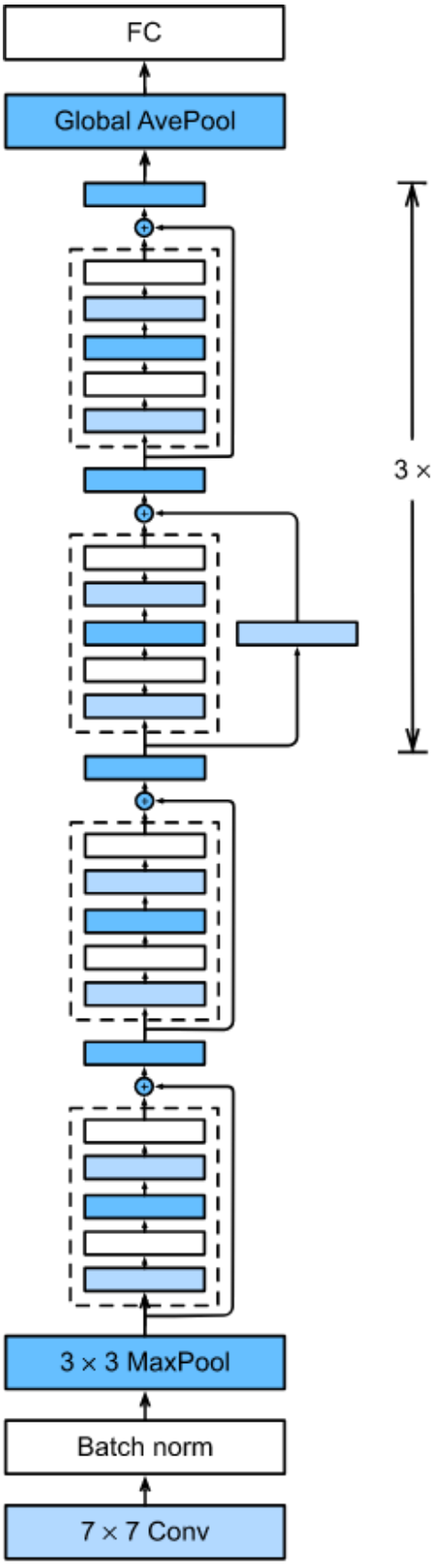
- Zero is padded with the skip connection to increase its dimensions.
- 1×1 convolutional layers are added to the input to match the dimensions. In such a case, the output is:

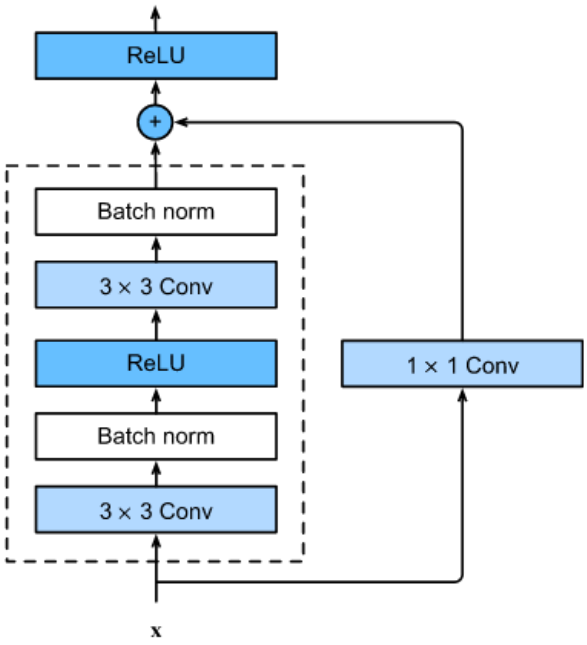
$$H(x)=f(x)+w1.x$$

Here an additional parameter w1 is added whereas no additional parameter is added when using the first approach.

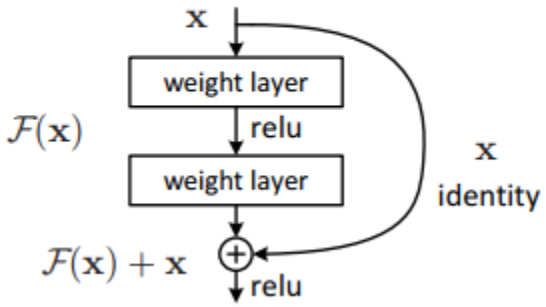
These skip connections technique in ResNet solves the problem of vanishing gradient in deep CNNs by allowing alternate shortcut path for the gradient to flow through. Also, the skip connection helps if any layer hurts the performance of architecture, then it will be skipped by regularization.

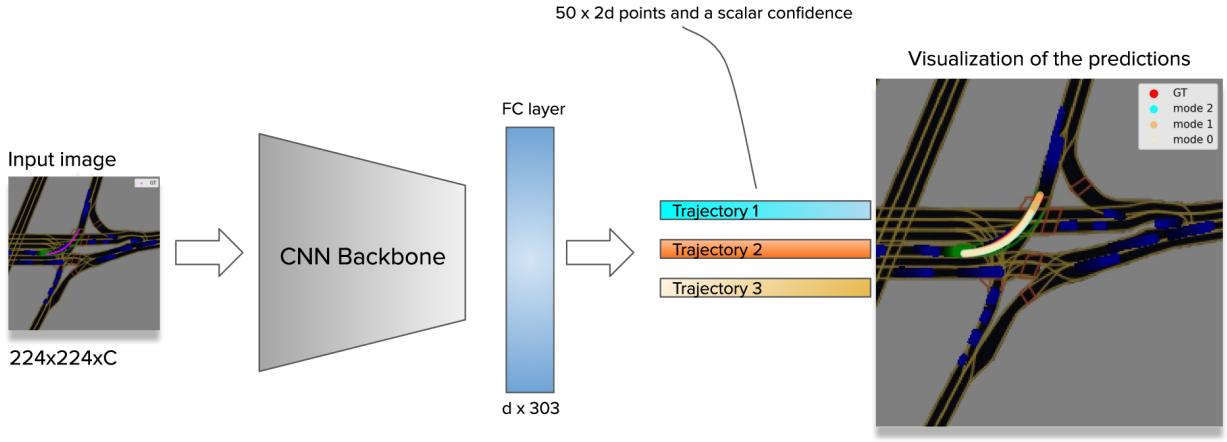
There are 4 convolutional layers in each module (excluding the 1×1 convolutional layer). Together with the first 7×7 convolutional layer and the final fully-connected layer, there are 18 layers in total. Therefore, this model is commonly known as ResNet-18. By configuring different numbers of channels and residual blocks in the module, we can create different ResNet models, such as the deeper 152-layer ResNet-152. Although the main architecture of ResNet is similar to that of GoogLeNet, ResNet's structure is simpler and easier to modify. All these factors have resulted in the rapid and widespread use of ResNet.The below figure depicts the full ResNet-18.





Residual block





B. Loss Function and Evaluation metrics

We are given a travel history of the agent and its current location on the map which was collected as we described in the previous section. For every agent we have from 0 to 50 history snapshots with a time interval of 0.1 seconds. So our input can be represented as an image with 3+(50+1)∗2=105 channels. Here the first 3 channels are the RGB map. Then we have 50 history time steps and one current. Every time step is represented by two channels: (1) The mask representing the location of the current agent, and (2) the mask representing all other agents nearby.

The test server allowed us to submit 3 hypotheses (proposals) for the future trajectory which were compared with the ground truth trajectory. The evaluation metric of this dataset - negative log-likelihood (NLL) of the ground truth coordinates in the distribution defined by the predicted proposals. The lower the better**.** In other words, given the ground truth trajectory **GT** and **K** predicted trajectory hypotheses $\mathbf{hypothesis_k}$**, k = 1,...,K** , we compute the likelihood of the ground truth trajectory under the mixture of Gaussians with the means equal to the predicted trajectories and the Identity matrix as a covariance.

Our goal is to predict three possible paths together with the confidence score, so we need to use the loss function that takes that into account, simply using RMSE will not lead to an accurate model.

Negative Log Likelihood (NLL): For each data sample in a multi-class classification task, NLL is calculated as:

$$NLL = -\sum_{c=1}^{M} y_c log(\hat{y}_c)$$

Where y_c is a binary indicator of correctness of predicting the data sample in class *c*, *y^c* is the predicted probability of the data sample belonging to class *c*, and *M* is the number of classes. Although NLL values are not as interpretable as previously discussed metrics, it can be used to compare the uncertainty of different intention prediction models.

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

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

which results in the loss

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

Computation Time: The trajectory prediction models are usually more complex compared to intention prediction models. Therefore, they can take more computation time which might make them impractical for on-board implementation in autonomous vehicles. Thus, it is crucial to report and compare computation time in trajectory prediction models.

Average Prediction Time: This metric is used in intention prediction approaches such as lane change prediction, where the approach is applied on a sliding window of the input data series to predict the occurrence of a positive class(e.g., lane change). The metric is obtained by taking the average of the time of the first correct positive class prediction for all samples, considering the time of lane change occurrence as the origin. We considered the time when a consistent correct lane change prediction starts to increase robustness of the metric.

4. Results

We employ our dataset for the SDV planning task. The task here is to predict and also execute a trajectory for the SDV. This allows the actual SDV pose to be different in future timesteps than recorded in the log. Using motion forecasting is not a suitable solution for this problem as it suffers from accumulation of errors due to broken i.i.d assumption. Our implementation is based on augmenting motion forecasting model with synthetic perturbations to mitigate this problem. The network is trained to accept BEV rasters of size 224 × 224 pixels (centered around the SDV this time) to predict future (x,y)-positions over a 5-second-horizon. We use the same architecture and loss as in the motion forecasting task, and train for a similar number of iterations. Testing of the system constitutes simulating SDV behaviour in 25 sec. long scenes from the test dataset. In each episode each traffic participant follows logged behaviour. The SDV is, however, controlled by the network and is free to take any action and diverge from its original behaviour. Results are summarised in Figure 8. This simulation is not perfect as it is non-reactive, but still gives valuable insights into SDV's actual performance. We count only errors caused by the SDV's actions (collisions, traffic rules violations, off-road events) and not limitations of blind log-replay simulation (e.g. other cars colliding into the SDV due to non-reactivity, diverging too far from the log position rendering perception ineffective), which dominate the total errors. Similarly to motion forecasting the performance keeps increasing with more data. This suggests that the performance is not saturated and both tasks can benefit from even larger datasets in the future.
In this thesis, motion prediction based on the road geometry and interaction with other vehicles is attempted. A major drawback of current state-of-the-art approaches is their inability to include information regarding both the interaction among vehicles, and the shape of the road. This limitation has been addressed by proposing new modeling techniques to can include road-geometric information and are compatible with some important existing models. A focus of this compatibility is with interaction-based predictions methods, due to the importance of both interaction-aware and road-aware prediction. To show this importance in modeling, 5 different models are constructed that vary in their road-geometric and interaction awareness. A baseline deep learning model is used and extended with a road-geometry module, an interaction module, a combination of the two, or none. Because the existing road-geometric models are inadequate, two new approaches have been designed that exploit map information of the scene. The prediction performances from the models clearly show the importance of both interaction-aware and road-geometry aware modeling. The road-refinement and road-attention models outperform a road-agnostic model on the data, showing better understanding of the road layout. The model utilizes reliable map information to assess

the shape of the road ahead of the vehicle, and shows that it can accurately predict in situations with different road layouts. Meanwhile, the model is capable of understanding the interaction between vehicles within these road scenarios. Therefore, with the road-attention module, a model is obtained that overcomes some of the most important limitations in current motion prediction approaches.
The goal of this competition is to predict the trajectories of other traffic participants. You can employ uni-modal models yielding a single prediction per sample, or multi-modal ones generating multiple hypotheses (up to 3) - further described by a confidence vector. Due to the high amount of multi-modality and ambiguity in traffic scenes, the used evaluation metric to score this model is tailored to account for multiple predictions. Every agent is identified by its `track_id` and its `timestamp`. Each trajectory holds 50 2D `(X,Y)` predictions.

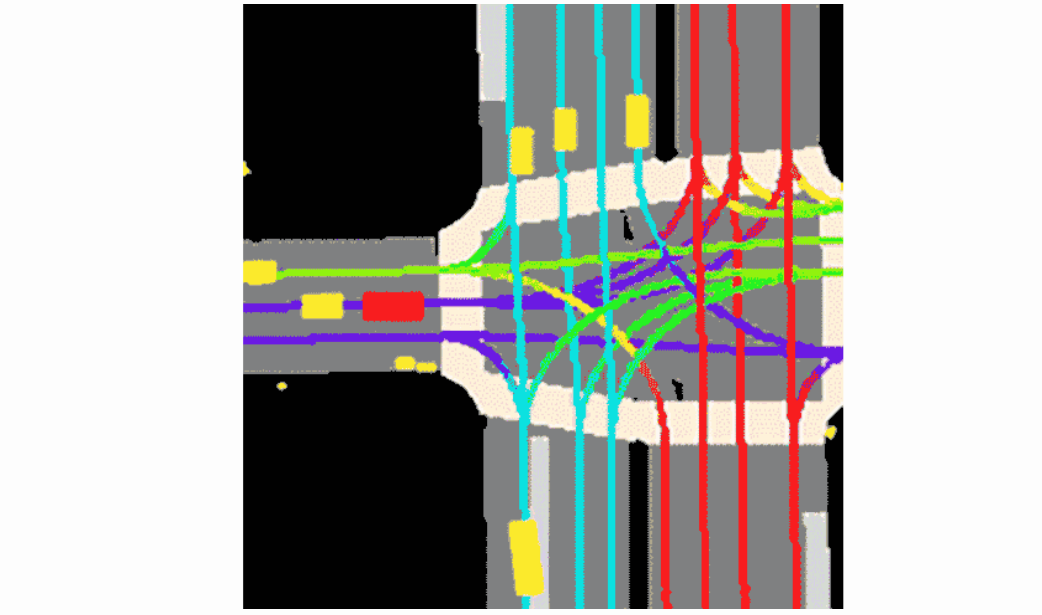
We predict up to 3 trajectories for each agent in the test set. Because the format is a CSV file, all 3 trajectories fields must have a value, even if your prediction is single-modal. However, each one of the three trajectory has its own confidence, and you can set it 0 to completely ignore one or more trajectories during evaluation. The 3 confidences must sum to 1.

We would like to discuss several limitations of our model. First, the hyperparameters associated with observed frame lengths (currently 10) and prediction frame lengths (current 5) may limit the capacity of our model. For example, if we only observe three frames, the model may fail to predict well on the following frames due to the lack of information. This means that the current model only allows the input length of 10. Second, our model is trained only using front-camera images from Waymo cars and may not be easily generalized to other car models whose cameras are mounted with different heights and angles.

5. Conclusion

From the experimental results, it is observed that the Resnet based model has better performance than the baseline models, which also supports our assumption.This work can be extended in several ways. One could LSTM-based model with an image has significant improvement. Through which we can accurately do motion planning with acceleration vectors. To further increase the model performance, camera images are introduced to help the model output a more accurate result. The result shows that the LSTM-based model with image has significant improvement. However, the model using the front camera only shares similar performance with the model using all five cameras while the latter took much longer time to train. Therefore, we conclude that the LSTM-based model with the front camera images is an effective and efficient model for

acceleration prediction.One limitation of the experiments we carried out is that, in the current training and testing process, the “online approach” takes the ground truth relative distance and velocities as input to the network after predicting the first frame after the initial 10 frames. However, in the scenario of the real world, the input should be taken from (or accumulated from) the prediction of previous frames. Therefore, our current approach may cause the loss to accumulate continually. To solve this problem, our future work is to generate predictions of one frame immediately after the initial ten frames, and then compute the loss between that specific frame and the ground truth data. Another future direction is to train a more generalizable model for other traffic scenarios than car-following—for example, lane-changing, merging and diverging.



Example of the agent and its history. In this case, the agent is an autonomous vehicle itself, but the dataset contains similar recordings for other cars as well. Colored lines depict road lanes.