Self-Driving Vehicle Traversing a Dynamic Environment

ENPM690 - Final Project Report

Aswath Muthuselvam
118286204
Master of Engineering in Robotics
University of Maryland
College Park, USA
aswath@umd.edu

Shailesh Pranav Rajendran
118261997
Master of Engineering in Robotics
University of Maryland, College Park
College Park, USA
spraj@umd.edu

Abstract

Robust Motion prediction for self-driving vehicles(SDVs) is essential for the vehicle to navigate a traffic rich environments without collision. In this work, we use a Vision Transformer(ViT) to learn the motion prediction model. The model is trained on sensor data collected from real world. We evaluate the training in two methods, i) open loop with supervised learning and ii) closed loop with reinforcement learning. In supervised learning setup, we show that the loss decreases as the model learns, the final validation loss of the model is 1.8. In reinforcement learning setup, the model showed signs of increasing rewards for the number of iterations that we ran for [1].

Keywords: Vision Transformer, Reinforcement learning, Motion prediction

1. Introduction

SDVs are gaining popularity and has become an area of interest among researches and the public. SDV companies like Waymo[1], Lyft and Motional[2, 3] are fuelling research by sharing their Datasets publicly. In this project, we use Lyft's dataset and their python library - L5Kit, which includes a gym environment and libraries for processing camera and lidar information. With Lyft's perception dataset[4] which has lidar, camera and satellite images , and a prediction dataset[5] which is created form radar, lidar, and camera. The dataset contains semantic information coded in color image, rendered from birds eye view.

Application scenarios Learning a model can be useful when we cannot possibly fanthom a wide variety of safety constraints that needs to be included. Given a good model architecture, hidden patterns in agents motion and high dimensional relationships can be learnt from data. Specifically, the non-linear dynamics can be learnt. Even when unseen data is received, the model can generalize to it in online manner. This is not possible when explicitly modelling a vehicle's behaviour. Thus, SDVs in realworld traffic, application scenarios for such learning based motion prediction.

Our contribution In this work, we use the prediction dataset to learn the motion model of a SDV and make it navigate in dynamic environments. We use a Deep Neural Network(DNN) as our model, more specifically we use a Vision Transformer[6]. We also evaluate our work with Resnet [7] and use it as a baseline comparison.

2. Related Works

Resnet [8] is popular choice for image classification task. It uses skip connections to solve the vanishing gradients problem during backpropagation and also during forward pass, the model is able to use low level features in the deeper layers of the network.

Transformers[9] were first introduced in NLP community. It offered benefits such as parallel processing of data with contrast to sequential processing architectures such as RNN, LSTM or GRU.

Later, Vision Transformers[6] were introduced in vision community. ViT divides images into patches and run inference with MLP at the end of the network.

Supervised learning Some similar works that was used as reference for our project includes [8] which is a motion prediction model using ResNet. Another paper [10], gives an overview of the implementation of a similar network.

A joint supervision on Planning, prediction and detection for a SDV is done by[11]. They use CNN as the backbone, along with techniques such as Neural Motion planner to plan a safe trajectory considering the hard to predict motion of actors in the scene.

[1]Code is publicly available at https://github.com/aswathselvam/ENPM-690/tree/master/Final_Project

Reinforcement learning Similar work to ours is done by [12], they perform training of a ViT Deep-Q-Network. They show promising results for few variants of ViT backbone compared to ResNet 18 and ResNet 50.

A CNN based occupancy network is proposed[13] and their problem formulation closely relates imitation and inverse reinforcement learning. They process raw Lidar information and the image map using a two stream network to predict the trajectory of a SDV.

3. Methodology

We use Deep neural networks as our motion prediction models. The models we use are Resnet and Vision Transformers.

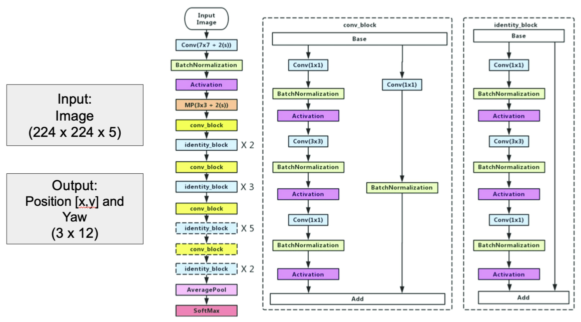


Figure 1. **ResNET50 Architecture**[14]. Boxes in the left show our input and output sizes.

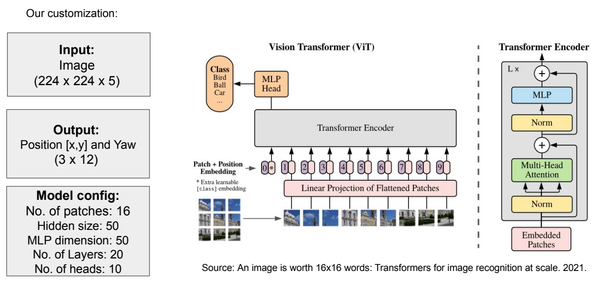
The CNN we use is residual neural network (ResNet). The architecture of resnet50 that we used is shown in 1. From the figure we can observe that on the left are blocks with dotted line represents modules that might be removed. On the middle is the Convolution block which changes the dimension of the input. On the right is the Identity block which will not change the dimension of the input.

Building a model using ResNet CNN architecture variants by stacking historical BEV (bird's-eye-view) of the agent history. ResNet initial weights are loaded and allowed to update the model during training.

We also use a Vision Transformer to learn the model. Vision Transformer has attention mechanism incorporated in it's architecture. The images are sliced into patches and the each patch is parsed parallelly with contrast to sequential processing of data done by RRNs.

3.1. Supervised Learning

Supervised learning teaches a model to predict the desired output by using the training data-set. The data-set consists of inputs and correct outputs which allows the model to learn over time. For our purpose we are using a supervised deep learning algorithm, Convolution Neural Network.



Our customization:
Input: Image (224 x 224 x 5)
Output: Position [x,y] and Yaw (3 x 12)
Model config: No. of patches: 16 Hidden size: 50 MLP dimension: 50 No. of Layers: 20 No. of heads: 10
Vision Transformer (ViT)
Class Bird Ball Car ...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches
Source: An image is worth 16x16 words: Transformers for image recognition at scale. 2021.

Figure 2. **ViT Architecture**[6]. Boxes in the left show the input, output sized and the model configurations.

For both the DNN architecture, the input provided to the network is 224 x 224 x 5 BEV rasterized images centered around the several agents in the model. And the output from the model is the 12 future position (x,y) and orientation - yaw (θ). The evaluation of the model is performed on Mean Square Error loss by comparing the predicted future trajectories with the observed.

The optimizer used is Adam Optimizer with a learning rate of 10^{-4}

The loss L we used is calculated as Mean Squared Error between the predicted value x and the ground truth data y:

$$L = ||x_{pos} - y_{pos}||_2 + ||x_{yaw} - y_{yaw}||_2 \tag{1}$$

3.2. Reinforcement Learning

L5kit returns the reward based on the environment. We found out that the reward is calculated as the L2 norm of sum of difference between the ground truth position and orientation data at that timestep and the position and orientation of simulated ego vehicle in the gym environment. The reward r generated by the environment is given by the following formula:

$$r = -||x_{pos} - y_{pos}||_2 - ||x_{yaw} - y_{yaw}||_2$$

We formulate the loss function for our model with various RL algorithms: i)Direct feedback, Q Learning, SARSA, and Bellman equation. However, we were only able to run the training for Direct feedback and Q learning.

Direct feedback

$$L_t = Q(s_t, a_t) \tag{2}$$

Q Learning

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_{t+1} + \gamma * max_a[Q(s_t, a_t)] - Q(s_t, a_t)) \tag{3}$$

SARSA

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_{t+1} + \gamma * Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \tag{4}$$

Bellman equation

$$V^*(s) = max_a(R(s,a) + \gamma \sum P(s_{t+1}|s,a)V^*(s_{t+1})) \tag{5}$$

The idea behind using Q Learning is that, the model from a previous iteration is used to pick the best action at the current timestep. The $max_a[Q(s_t, a_t)]$ operation is done by the substitution of the previous iteration model's weights to the target model.

The reward γ we chose for our model is 0.7, and the discount factor α we chose was 0.6. Higher learning rate increases.

We use Adam optimizer, with a learning rate of 10^{-3}. We used a batch size of 12, and ran it for 40000 iterations.

4. Simulation

The simulation environment used is OpenAI gym environment customized by Lyft. Our task builds on top of the agent behavior to predict the trajectories in the future time period. Deep learning solution leverages the birds-eye-view(BEV) representation of the agent movement represented as a semantic map with lanes, agents, obstacles and other information observed by AV's lidars, radars and cameras. Our goal from each run of the ego is for it to avoid collision as it moves along the highway following the lane, stopping when the traffic signal is red, braking and continuing in an intersection and turning to a perpendicular road in the intersection.

4.1. Data-set

The data-set that we are using is the largest self-driving data-set for motion prediction to date, with over 1,000 hours of data. This was collected by a fleet of 20 autonomous vehicles along a fixed route in Palo Alto, California over a four-month period. It consists of 170,000 scenes, where each scene is 25 seconds long and captures the perception output of the self-driving system, which encodes the precise positions and motions of nearby vehicles, cyclists, and pedestrians over time. On top of this, the data-set contains a high-definition semantic map with 15,242 labelled elements and a high-definition aerial view over the area. Together with the provided soft- ware kit, this collection forms the largest, most complete and detailed data-set to date for the development of self- driving, machine learning tasks such as motion forecasting, planning and simulation.

The data-set has three components:

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

The planning dataset contains Road Graph Layer, Lane Geometry Layer, Semantic Features and Map Priors. Each of these layers will be described in detail below. With each of these layers, a final image is rendered and used as the observation for learning the model. The image is of size `112x112x7`.

Road Graph Layer At the foundation of our semantic map is the road network graph. This represents all of the road segments and the interconnections: how many lanes there are, what direction they travel, and which roads connect to which. It also represents the yield relationships between roads and lanes, so that the AVs are able to safely stop at intersections or crosswalks for cross traffic.

Lane Geometry Layer Above the road graph is a centimeter-precise lane geometry. This is a set of polygons that represents the individual lane markings on the road surface and the street-level rules the vehicle will obey. This data also contains properties not usually found in navigational maps: color of lines, areas that allow lane changes, speed bumps, and stop lines are all represented.

Semantic Features At the top levels are the semantic features and map priors. Semantic features are traffic lights, crosswalks, and road signs. Map priors are areas in the map that we expect to see something with a correlating probability. These two aspects of the map allow for the vehicle to make decisions about how it and other objects will behave.

Map Priors Map Priors are similar but allow even more nuance. These represent derived or observed information that is encoded into the map. Using the case of traffic lights, the prior layer contains the order that an individual traffic light fixture cycles through each state (red, green arrow, green, yellow, then red again) and how long each state lasts. An example for differentiating between semantic features and map priors is parking. As a semantic feature, a parking spot designates an area where the vehicle can't drive and that cars must come to a stop. As a prior, parking may represent an area of the street that is safe to drive on, yet also indicates that parked cars have been seen in the area before.

Software Packages

- Programming language: Python 3.8
- Dependancies: numpy, Opencv, matplotlib, pytorch, bokeh, tensorboards, gym, zarr

5. Experiment Results

Supervised Learning results We show the train and validation loss plots for Supervised Learning. For the plots, the learning rate was 10^{-4}, ViT model with hidden size 50, MLP dimension of 1024, 10 heads and 16 layers. We ran the model for 13000 iterations. Note that the losses are plotted every 50 iterations.

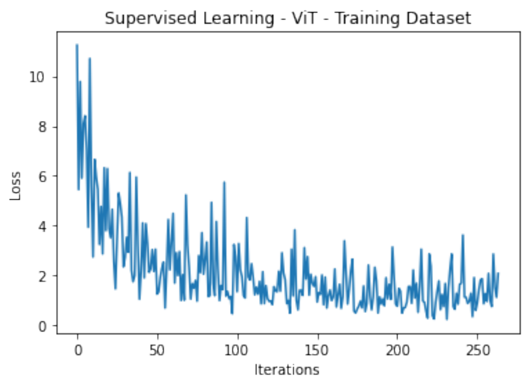


Figure 3. **Train loss** of supervised Learning with ViT model. Loss decreases continuously.

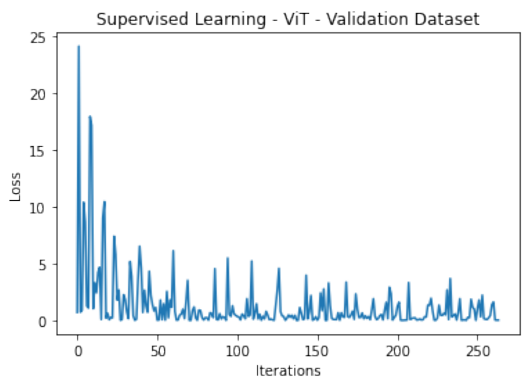


Figure 4. **Validation loss** of supervised Learning with ViT model. Loss decreases continuously

Models	Val. Loss	lr = 10^{-3}	lr = 10^{-4}
ResNet18	Min. Loss	1.23	1.66
	Avg. Loss	1.79	2.21
ResNet50	Min. Loss	0.986	1.24
	Avg. Loss	1.57	1.74
ViT	Min. Loss	1.15	1.17
	Avg. Loss	1.41	1.63

Table 1. **Loss Plot for Supervised Learning** Blue numbers show the best model performance.

We plot the attention map of the vision transformer to understand what the model has learnt and which parts of the BEV image, the attention is being made. Shown in Figure 6, the attention is being made around the vehicle which is very close behind the ego vehicle. Just a few time steps later, as shown in Figure 6, the attention is shifted to the T junction as the ego vehicle approaches closer to the intersection.

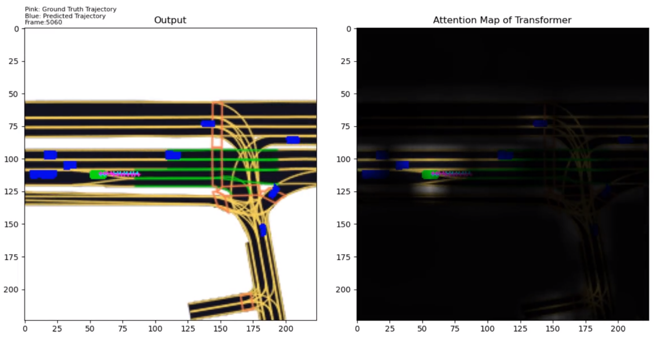

Pink: Ground Truth Trajectory
Blue: Predicted Trajectory
Frame:5060
Output
Attention Map of Transformer

Figure 5. **Attention Map before intersection**. Only the area surrounding the ego-vehicle is bright.

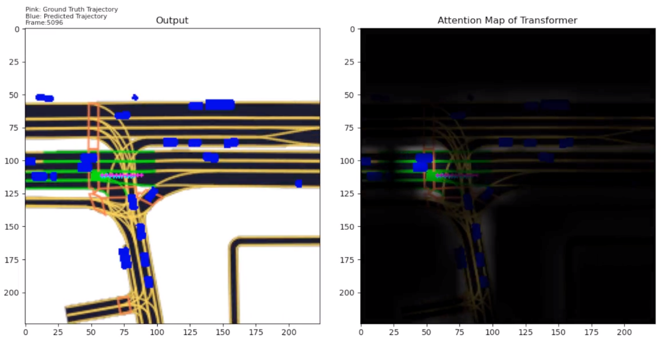

Pink: Ground Truth Trajectory
Blue: Predicted Trajectory
Frame:5096
Output
Attention Map of Transformer

Figure 6. **Attention Map near intersection**. The vertical road at the bottom part of the image is brighter.

Reinforcement learning results For reinforcement learning we have used the ViT model to learn using the Q-Learning algorithm. We trained the model with various configurations by tuning the parameters such as the learning rate of the model, learning rate γ of the Q Learning algorithm, and model parameters like MLP dimension, head size, and the number of layers.

Configuration	Min. Loss	Avg. loss
10K params.	-4.5	-2.35
27K params.	-2.34	-1.9
learning rate 10^{-3}	-3.1	-6.1
learning rate 10^{-4}	-4.5	-6.6

Table 2. Reinforcement Learning with ViT backbone. Blue numbers show the lowest loss.

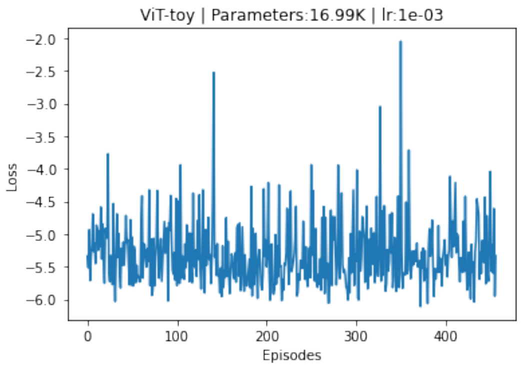


Figure 7. **Loss Plot for Reinforcement Learning** At 35000 iterations, we see the lowest loss value.

Even though, we did not see any trends for convergence in the loss, we did see a very low loss value of -2.34 towards the end of the training at 35000 iterations. Table 2 shows the loss values obtained for different configurations. The base configuration used is as follows: MLP dimension is 50, number of heads is 10, number of layers is 5.

6. Conclusion and Future Work

In this project, we presented the learning of motion-prediction model of SDVs using real world traffic data including scenarios such as highways and intersections. We demonstrated that the model learns to identify context from the situations. In reinforcement learning setup, the reward function could include heuristic information to make the model learn faster. The field of Autonomous vehicles are young and bright.

We identified plenty of scope for improvements from this work. At the moment, physics informed Neural Networks are gaining traction. We can see the traffic as an interconnected dynamical system and use a PINN to jointly model the higher order derivatives of position, this can reduce the number of parameters of the backbone DNN and the ego SDV has more robust prediction. Temporal models like LSTM, RNN can improve the accuracy of motion prediction as it predicts based on the history of states.

References

[1] Waymo open dataset: An autonomous driving dataset, 2019. 1

[2] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*, 2021. 1

[3] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1

[4] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Level 5 perception dataset 2020. https://level-5.global/level5/data/, 2019. 1

[5] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Long Chen, 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. 1

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[8] Yao ZeHao, LiQian Wang, Ke Liu, and YuanQing Li. Motion prediction for autonomous vehicles using resnet-based model. In *2021 2nd International Conference on Education, Knowledge and Information Management (ICEKIM)*, pages 323–327, 2021. doi: 10.1109/ICEKIM52309.2021.00078. 1

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[10] Imre Fodi. Building motion prediction models for self-driving vehicles. 1

[11] Wenyuan Zeng, Shenlong Wang, Renjie Liao, Yun Chen, Bin Yang, and Raquel Urtasun. Dsdnet: Deep structured self-driving network. In *European conference on computer vision*, pages 156–172. Springer, 2020. 1

[12] Eshagh Kargar and Ville Kyrki. Vision transformer for learning driving policies in complex multi-agent environments. *arXiv preprint arXiv:2109.06514*, 2021. 2

[13] Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *European Conference on Computer Vision*, pages 414–430. Springer, 2020. 2

[14] Qingge Ji, Jie Huang, Wenjie He, and Yankui Sun. Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images. *Algorithms*, 12:51, 02 2019. doi: 10.3390/a12030051. 2