Motion Prediction for Autonomous Vehicles

Sabbisetti Sri Sai Keerthi
Department in Electronics and Communication Engineering
Vellore Institute of Technology
Vellore,Tamilnadu,India

Jonnagadla Bhava Samhitha
Department in Electronics and Communication Engineering
Vellore Institute of Technology
Vellore,Tamilnadu,India

Sankar Ganesh S
Department of Communication Engineering (Assistant Professor Sr- Grade 1)
Vellore Institute of Technology
Vellore,Tamilnadu,India

***Abstract*—The advent of autonomous vehicles is poised to revolutionize the transportation industry across the globe. However, developing self-driving cars comes with various technical hurdles that need to be addressed. One major challenge is creating accurate models that can predict the movements of traffic participants such as pedestrians, cyclists, and other cars around the self-driving cars. The purpose of this research is to assess the effectiveness of different deep learning models in predicting such movements by evaluating their root mean square error score. These deep learning models leverage the current state of the surrounding environment to forecast the motion of traffic agents.**

***Index Terms*——Autonomous Vehicles (AV), Deep Learning (DL), Artificial Intelligence (AI)**

I. Introduction

Driving a motor vehicle is a complex task that requires drivers to analyze and consider the movements of peripheral agents such as pedestrians, cyclists, and other vehicles before making a move. Unfortunately, humans are not well-suited for this task, as evidenced by the fact that traffic accidents are one of the leading causes of death and injury worldwide. Research has shown that a significant percentage of these accidents are caused by human error. This has led to the development of autonomous vehicle (AV) technology, which is based on the idea of driverless cars. AVs make decisions based on input from external sensors like lidar and cameras, and the processing power of modern computers far exceeds human capabilities.

To safely deploy AVs on public roads, it is essential to analyze various tasks such as observing and predicting the motion and trajectories of peripheral agents and navigating the AV to the intended destination while taking into account the state of the agents. The main objective of this research is to train deep learning models using datasets and optimize the loss function for training and validation datasets. The research aims to predict the future coordinates of trajectories for peripheral agents and calculate the root mean square error (RMSE) score from the ground truth trajectory and the predicted trajectory for the future motion of the agents, as shown in Figure 1.

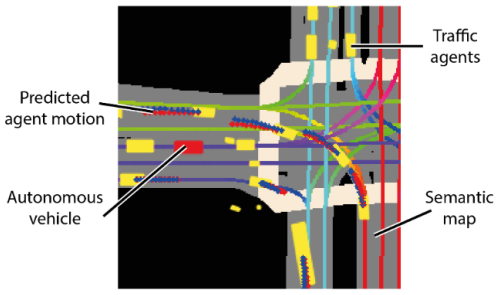


Fig. 1. "semantic map with ground agent motion and predicted agent motion"

II. Related Work

The development of autonomous driving technology relies heavily on the ability of the system to make decisions based on real-time information. Learning-based decision-making methods are a key component of this technology, and can be broadly divided into two categories: policy learning and model learning.

Policy learning involves training a policy, typically parameterized by neural networks, to output driving decisions based on the current state input. Two popular policy learning approaches are imitation learning (IL) and reinforcement learning (RL). IL seeks to imitate expert decisions from demonstrations, but suffers from the problem of distributional shift during online deployment, resulting in inferior testing performance. RL, on the other hand, learns online through interactions with the environment, which addresses the distributional shift issue, but can be very inefficient due to trial-and-error learning. Some model-based RL approaches attempt to build a transition model of dynamics and reward function, and use it to learn or improve a policy. Model learning, on the other hand, learns a model to predict environment dynamics and plan over the model, improving the explainability, robustness, and safety of the system compared to policy learning.

Motion prediction, another critical component of autonomous driving technology, involves predicting long-term

future motion trajectories of traffic participants based on their historical dynamic states and optionally the map information. Recent motion prediction networks leveraging Transformers or GNNs have achieved unprecedented prediction accuracy, but most focus solely on improving prediction accuracy, ignoring the applicability to downstream planning tasks. One key issue with existing models is that they are not aware of the AV's future plans, and the prediction results are not reactive to the AV's different decisions, forcing the AV to act passively. Some recent works have attempted to mitigate this issue, such as PiP, which proposes a planning-informed trajectory prediction network that conditions the prediction process on the candidate trajectories of the AV, and conditional behavior prediction, which formulates a framework for such a prediction model. However, these works are still largely focused on the prediction part, and less attention has been paid to decision-making performance and interactive behaviors.

To address these limitations, the authors propose an interaction-aware motion prediction model that can make accurate and reactive predictions of surrounding traffic agents to support the decision-making process. The proposed model leverages the framework of receding horizon control, which allows for multi-step look-ahead planning, and is trained online through interactions with other agents. The authors thoroughly evaluate the decision-making performance of the interactive prediction model, and find that it outperforms existing models on a range of metrics, including average speed, total travel time, and safety.

Overall, the authors' work represents an important contribution to the field of autonomous driving technology, addressing key limitations in existing learning-based decision-making and motion prediction models. The proposed interaction-aware model has the potential to significantly improve the safety, efficiency, and overall performance of autonomous driving systems.

III. Methods

The task at hand is to predict the movement of an object for the next T seconds, assuming that its tracks are already provided by a perception system. Our focus is solely on motion prediction, and we accomplish this by first converting the data into multi-channel images through a process called rasterization. We then proceed to explain the design of our model, as well as the loss function used to train it.

A. Rasterization

In order to create images for training, we convert the raw data by combining the past trajectories of the agents with a map that contextualizes the surrounding road environment. We take steps to standardize the input by shifting and rotating the frame so that the target agent is consistently situated at a fixed location on the raster image at the time of prediction, and ensuring that its velocity is aligned with the X-axis.

B. Model

The future is uncertain, our goal is to generate multiple possible trajectories for the future movement of the object, and evaluate each proposal against the actual trajectory. This baseline solution uses a ResNet18 model architecture to train on over 2 million samples from the dataset. The model predicts the future coordinates of a single agent at a time, based on a bird's eye view (BEV) top-down raster that encodes all agents and the map. This raster is generated by taking a sequence of ten consecutive frames (representing one second of data) as input and encoding it into a fixed-sized tensor. To train the model, a batch size of 16 is used over 30,000 iterations. The input size of the model is 300px, with a history of 1s. This means that the model takes in a sequence of ten frames, each of which is 300 pixels wide, to make a prediction about the movement of a traffic agent in the eleventh frame. The optimizer used is Adam with a learning rate of 1e-3. The loss function used is Mean Squared Error (MSE) loss.

C. Model Architecture

ResNet architecture: The model being used here is "ResNet", which stands for Residual Network which, as the name suggests supports Residual Learning. In conventional deep learning convolutional neural networks, multiple layers are stacked upon each other and trained accordingly. By contrast in residual learning, the network doesn't directly try to learn features but instead tries to learn some residual. Residual; to be understood easily can be described as to be subtraction of the features from the input layer of that particular layer. This is done by using shortcut connections between the layers (directly connecting the input of yth layer to the output of some (y + n)th layer). These are comparatively easier to implement and train than conventional models and also the common problem faced by all of degrading accuracy is massively resolved.

D. Loss Function

MSE (Mean Squared Error) is a popular metric used in machine learning to evaluate the accuracy of a model's predictions. It measures the average squared difference between the predicted and actual values, where larger errors are weighted more heavily. This metric is commonly used as a loss function during model training to minimize the difference between predicted and actual values.In addition, the MSE is a continuous measure, meaning that it can be used to compare the performance of different models in a meaningful way. This is important because it allows researchers to quantitatively evaluate the impact of different modeling choices, such as the choice of input features or the regularization parameter.However, MSE has some limitations, such as sensitivity to outliers and the fact that it is calculated using the square of the differences, which may not be the best metric when the absolute difference is more important. Despite these limitations, MSE is widely used due to its simplicity and effectiveness in evaluating the accuracy of a model's predictions, making it a valuable tool in machine learning and statistical modeling.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{1}$$

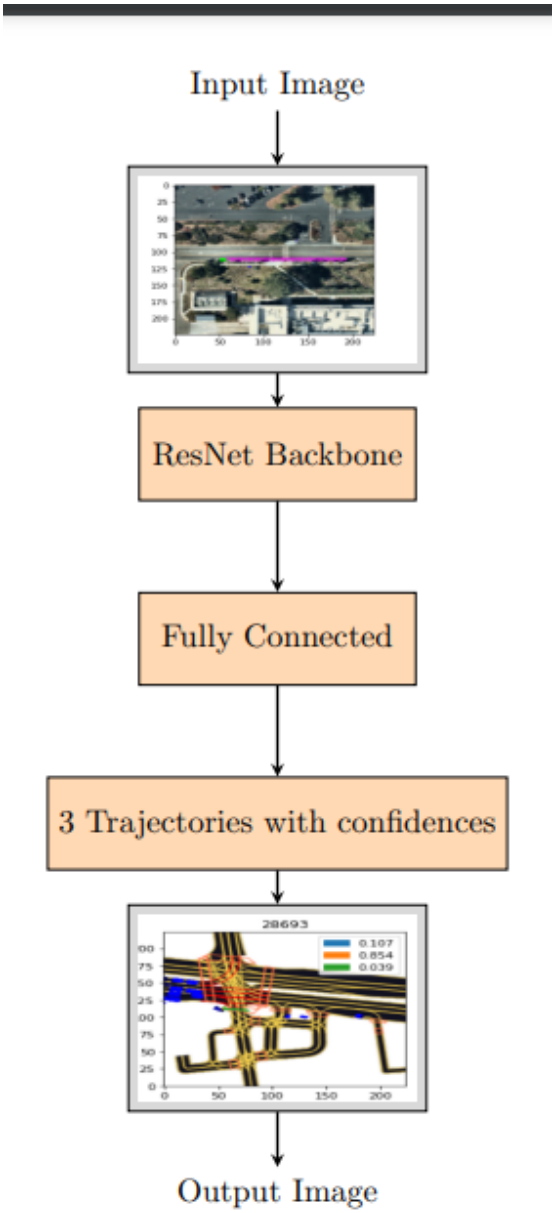


Fig. 2. "Overview architecture of our model"

IV. Experiments

A. Data set

The data set that has been released includes 170,000 scenes, with each scene capturing the movement of the self-driving vehicle, other traffic participants, and the state of traffic lights. The data set also includes a high-definition semantic map and a high-resolution aerial picture that can be used to enhance prediction. The scenes were captured by a fleet of self-driving vehicles using seven cameras, three LiDARs, and five radars. The LiDAR on the roof of the vehicle has 64 channels and spins at 10 Hz, while the two LiDARs on the front bumper have 40 channels. All seven cameras are mounted on the roof and have a 360° horizontal field of view. The four radars are also mounted on the roof, while one radar is placed on the forward-facing front bumper.

The data set contains over 1,118 hours of logs, with each scene being 25 seconds long, and the data set has been split into a training set, a validation set, and a test set. The training set includes 928 hours of logs and 134,000 scenes, while the validation set includes 78 hours of logs and 11,000 scenes, and the test set includes 112 hours of logs and 16,000 scenes. The total size of the data set is 1,118 hours, 26,344 km, and 162,000 scenes. The data set was collected between October 2019 and March 2020 during daytime, between 8 AM and 4 PM.

The data set provides precise information about traffic participants, including vehicles, pedestrians, and cyclists. Each traffic participant is represented by a 2.5D cuboid, velocity, acceleration, yaw, yaw rate, and a class label. The data set also provides information about the road itself, including lane geometry, road rules, and other traffic elements. The high-definition semantic map includes information about lane boundaries, lane connectivity, driving directions, road class, road paintings, speed limits, lane restrictions, crosswalks, traffic lights, traffic signs, restrictions, and speed bumps. The semantic map also includes precise road geometry, which can be used for planning driving behavior and anticipating the movements of other traffic participants. The semantic map is given in the form of a protocol buffer and includes a total of 15,242 labeled elements, including 8,505 lane segments.

The data set has been encoded in the form of n-dimensional compressed zarr arrays. The zarr format allows for fast random access to different portions of the data set while minimizing the memory footprint, which enables efficient distributed training on the cloud.

The data set includes a large number of scenes captured by a fleet of self-driving vehicles using a variety of sensors, providing precise information about traffic participants and the road itself. The data set also includes a high-definition semantic map and a high-resolution aerial picture, which can be used to enhance prediction. The data set has been split into a training set, a validation set, and a test set and has been encoded in the form of n-dimensional compressed zarr arrays, enabling efficient distributed training on the cloud.

B. Implementation Details

The focus of this project is to develop and evaluate motion prediction models using the Lyft Motion Prediction for Autonomous Vehicles Kaggle competition dataset. The aim is to predict the movements of traffic agents such as cyclists, pedestrians, and cars around autonomous vehicles. The approach taken is to build on the single mode baseline solution provided by the competition organizers, which uses a ResNet18 model architecture to train on over 2 million samples from the dataset.

The model predicts the future coordinates of a single agent at a time based on a bird's eye view (BEV) top-down raster that encodes all agents and the map.

To implement this approach, the primary programming language used is Python, and the PyTorch deep learning framework is used for building and training the models. Additionally, the Lyft L5Kit library is used to provide a toolkit for working with the Lyft Level 5 AV data set. The design follows industry best practices for deep learning model development, including good code organization, version control using Git, and modular design. The constraints, alternatives, and trade offs considered include selecting appropriate hyper parameters and balancing model accuracy with computational complexity. The ResNet18 architecture is chosen for its balance between accuracy and computational efficiency, although other models like ResNet50 or DenseNet may offer higher accuracy at the cost of increased computational complexity. The goal is to optimize the model's performance without over fitting the training data.

Balancing model accuracy with computational complexity is another important trade-off. The ResNet18 architecture is chosen for its balance between accuracy and computational efficiency. Other models like ResNet50 or DenseNet may offer higher accuracy at the cost of increased computational complexity. The goal of this project is to optimize the model's performance without overfitting the training data.

C. Metrics

The negative log-likelihood (NLL) of a mixture of Gaussians can be used as the loss function to evaluate the predicted trajectory hypotheses. The NLL is computed by taking the ground truth trajectory and finding the negative log probability of it under the predicted mixture of Gaussians, where the means are equal to the predicted trajectories and the identity matrix I is used as the covariance. The loss function can be further decomposed into the product of 1-dimensional Gaussians. Although the proposed loss function does not explicitly penalize the model for generating similar trajectories, the model is not at risk of mode collapse because combining all the probability mass into one mode leads to a higher loss value in case of a misprediction. Therefore, optimizing the proposed loss function results in sufficient multimodality.In other words, given a ground truth trajectory

$$\text{L = -log } P(X_{\text{gt}}) = -\log\left(\sum_{k=1}^{K} c_k \mathcal{N}(X_{\text{gt}}; \mu_k, I)\right) \quad (2)$$

and K predicted trajectory hypotheses

$$\text{X}_k = [(x_{k,1}, y_{k,1}), \ldots, (x_{k,T}, y_{k,T})], \quad k = 1, \ldots, K \quad (3)$$

we compute negative log probability of the ground truth trajectory under the predicted mixture of Gaussians with the means equal to the predicted trajectories and the identity matrix I as covariance:

$$\text{L=-log } \mathcal{L}(\theta|x) = -\sum_{i=1}^{n} \log f(x_i|\theta) \quad (4)$$

where N (·; μ, sigma) is the probability density function for the multivariate Gaussian distribution with mean μ and covariance matrix sigma. The loss can be further decomposed into the product of 1-dimensional Gaussians, and we get just a logarithm of the sum of the exponents:

$$\text{L = -log } P(X_{\text{gt}}) = -\log\left(\sum_{k=1}^{K} c_k \mathcal{N}(X_{\text{gt}}; \mu_k, I)\right) \quad (5)$$

The proposed loss function does not have a direct penalty for the model producing very similar trajectories. However, based on empirical observations, we have not encountered a scenario where all probability is assigned to a single mode resulting in higher risk and loss values in case of incorrect predictions. Hence, optimizing the proposed loss is adequate to achieve sufficient multimodality.

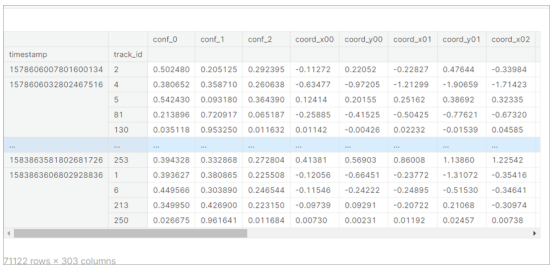
D. Results

Our first model is a single mode baseline that uses the resnet18 architecture, trained for 30000 iterations with a batch size of 16. The input size for this model is 300px with a history of 1 second or 10 frames. The optimizer used for this model is Adam with a learning rate of 1e-3, and the loss function used is Mean Squared Error (MSE) Loss. The Lower Bound (LB) score for this model is 246.349.

Our second model is also a single mode baseline using the resnet18 architecture, trained for 30000 iterations with a batch size of 16. However, the input size for this model is 350px with a history of 1 second or 10 frames. The optimizer used for this model is Adam with a learning rate of 1e-3, and the loss function used is MSE Loss. The LB score for this model is 169.83.

These results suggest that the second model with an input size of 350px performs better than the first model with an input size of 300px. The LB score of the second model is significantly lower, indicating better performance in predicting the trajectory of the vehicles. The use of the resnet18 architecture and the Adam optimizer with a learning rate of 1e-3 remained the same for both models, but the change in input size seemed to have a significant impact on the performance of the model.Further research and experimentation are necessary to determine the optimal approach for predicting vehicle trajectories in autonomous driving systems.

3 curves corresponding to 3 mode of predictions. The legend indicates the confidence scores. The bright green is history.



		conf_0	conf_1	conf_2	coord_x00	coord_y00	coord_x01	coord_y01	coord_x02
timestamp	track_id								
1578606007801600134	2	0.502480	0.205125	0.292395	-0.11272	0.22052	-0.22827	0.47644	-0.33984
1578606032802467516	4	0.380652	0.358710	0.260638	-0.63477	-0.97205	-1.21299	-1.90659	-1.71423
	5	0.542430	0.093180	0.364390	0.12414	0.20155	0.25162	0.38692	0.32335
	81	0.213896	0.720917	0.065187	-0.25885	-0.41525	-0.50425	-0.77621	-0.67320
	130	0.035118	0.953250	0.011632	0.01142	-0.00426	0.02232	-0.01539	0.04585
...	...	...	...	...	...	...	...	...	...
1583863581802681726	253	0.394328	0.332868	0.272804	0.41381	0.56903	0.86008	1.13860	1.22542
1583863606802928836	1	0.393627	0.380865	0.225508	-0.12056	-0.66451	-0.23772	-1.31072	-0.35416
	6	0.449566	0.303890	0.246544	-0.11546	-0.24222	-0.24895	-0.51530	-0.34641
	213	0.349950	0.426900	0.223150	-0.09739	0.09291	-0.20722	0.21068	-0.30974
	250	0.026675	0.961641	0.011684	0.00730	0.00231	0.01192	0.02457	0.00738

71122 rows × 303 columns

Fig. 3. ”Display of output file”

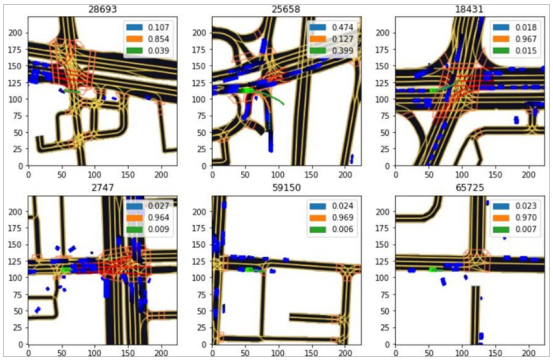


Fig. 4. ”Predicting the trajectory of a vehicle”

CONCLUSIONS

The Lyft dataset used in this paper is currently the most extensive and detailed public dataset available for training solutions, surpassing even the best alternative by three times in size and descriptive quality. The results of the study indicate that an increase in the number of parameters leads to better model performance, as evidenced by a decrease in the ADE value. This is because more layers in the model allow for the extraction of additional features, and an increase in parameters helps to fine-tune the model. Overall, the paper aims to contribute to future research efforts aimed at improving the performance of self-driving vehicles.

Fig. 5. ”Predicting the trajectory of a vehicle from satellite view”

REFERENCES

[1] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, Nitin Singh, Jeff Schneider. Uncertainty-aware Short-term Motion Prediction of Traffic Actors for Autonomous Driving. IEEEXplore 2020.

[2] NCHS. Health, United States, 2016: With chartbook on long-term trends in health. Technical Report 1232, National Center for Health Statistics, May 2017.

[3] NHTSA. Early estimate of motor vehicle traffic fatalities for the first half (jan–jun) of 2017. Technical Report DOT HS 812 453, National Highway Traffic Safety Administration, December 2017.

[4] S. Singh. Critical reasons for crashes investigated in the national motor vehicle crash causation survey. Technical Report DOT HS 812 115, National Highway Traffic Safety Administration, February 2015.

[5] L. J. Blincoe, T. R. Miller, E. Zaloshnja, and B. A. Lawrence. The economic and societal impact of motor vehicle crashes, 2010 .

[6] 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. Jun 2020.

[7] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. Int. Journal of Robotics Research (IJRR), 2013

[8] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 336–345, 2017.

[9] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In Int. Conf. Learn. Represent. (ICLR), 2017.

[10] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y.Zhang, J. Shlens, Z. Chen, and D. Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. 2019.

[11] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang. The apolloscape open dataset for autonomous driving and its application. Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2019.

[12] M. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, and J. Hays. Argoverse: 3d tracking and forecasting with rich maps. Int. Conf. on Computer Vision and Pattern Recognition (CVPR), 2019

[13] Qizhao Fang, Rui Hu, Xiangpeng Li, and Kristen Grauman. Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout

[14] Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 14074–14083, 2020.

[15] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In 2019 International Conference on Robotics and Automation (ICRA), pages 2090–2096. IEEE, 2019.

[16] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. Tnt: Target-driven trajectory

[17] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Sheng Zhao, Shuyang Cheng, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset, 2020 prediction. arXiv preprint arXiv:2008.08294, 2020

[18] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Sheng Zhao, Shuyang Cheng, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset, 2020

[19] Naga Srinivasu, P., Balas, V.E., Md. Norwawi, N., “Performance measurement of various hybridized kernels for noise normalization and

enhancement in high-resolution MR images", Studies in computational Intelligence 2021

[20] Fang-Chieh Chou, Tsung-Han Lin, Henggang Cui, Vladan Radosavljevic, Thi Nguyen, Tzu-Kuo Huang, Matthew Niedoba, Jeff Schneider and Nemanja Djuric. "Predicting Motion of Vulnerable Road Users using High-Definition Maps and Efficient ConvNets" 2021

[21] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider and Nemanja Djuric. "Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks" 2020

[22] Peiliang Li, Xiaozhi Chen, Shaojie Shen. "Stereo R-CNN based 3D Object Detection for Autonomous Driving" 2022.

[23] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, Xiaogang Wang. "An Efficient 3D Object Detection Framework for Autonomous Driving" 2020.

[24] A. Alahi, K. Goel, V. amanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. Int. Conf. on Computer Vision and Pattern Recognition (CVPR), 2016.

[25] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In Int. Conf. on Computer Vision and Pattern Recognition, 2018.

[26] Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun. Multitask multi sensor fusion for 3d object detection. Int. Conf. on Computer Vision and Pattern Recognition, 2019