Improving Motion Forecasting for Autonomous Driving with the Cycle Consistency Loss

Titas Chakraborty[*]
Carnegie Mellon University
titas0602@gmail.com

Akshay Bhagat
Motional
akshay.bhagat@motional.com

Henggang Cui
Motional
henggang.cui@motional.com

Abstract

Robust motion forecasting of the dynamic scene is a critical component of an autonomous vehicle. It is a challenging problem due to the heterogeneity in the scene and the inherent uncertainties in the problem. To improve the accuracy of motion forecasting, in this work, we identify a new consistency constraint in this task, that is *an agent's future trajectory should be coherent with its history observations and visa versa*. To leverage this property, we propose a novel cycle consistency training scheme and define a novel cycle loss to encourage this consistency. In particular, we reverse the predicted future trajectory backward in time and feed it back into the prediction model to predict the history and compute the loss as an additional cycle loss term. Through our experiments on the Argoverse dataset, we demonstrate that cycle loss can improve the performance of competitive motion forecasting models.

1 Introduction

Motion forecasting in autonomous vehicles is a challenging research problem to solve for the autonomous driving industry. Motion forecasting involves predicting the future trajectories of the agents in a scene given their history trajectories and the scene context, usually in the form of a lane graph. The problem is multi-modal since, given a history, there can be multiple possible futures. For example, at an intersection, an agent can take different possible maneuvers (e.g., straight, right turn, left turn) and follow different speed profiles.

Motion forecasting approaches are highly diverse in their input representations. Several methods, such as [4, 1], found success representing the input scene as an image and using convolutional neural networks to generate a scene representation. More recently, vectorized representations used in [17, 7, 11, 10] have found success due to computationally efficient sparse representation and ability to model long-range dependencies. [9, 4, 5, 6, 15] represent lanes as nodes in a graph to make use of the connectivity information. There are several output representations and loss functions in use as well. [14, 9] predict an unconstrained regressed output and use a simple regression loss. [4, 5, 6] predict an unconstrained heatmap with a cross entropy loss, sample the trajectory endpoint from the heatmap, and they complete the whole trajectory conditioned on the endpoint.

There are a number of consistencies and symmetries inherent to the motion forecasting problem. Recently, there has been an effort to integrate this information, either as an explicit loss or in the model structure to improve the model performance. Certain symmetries such as rotation, translation

[*]Work done while interning at Motional.

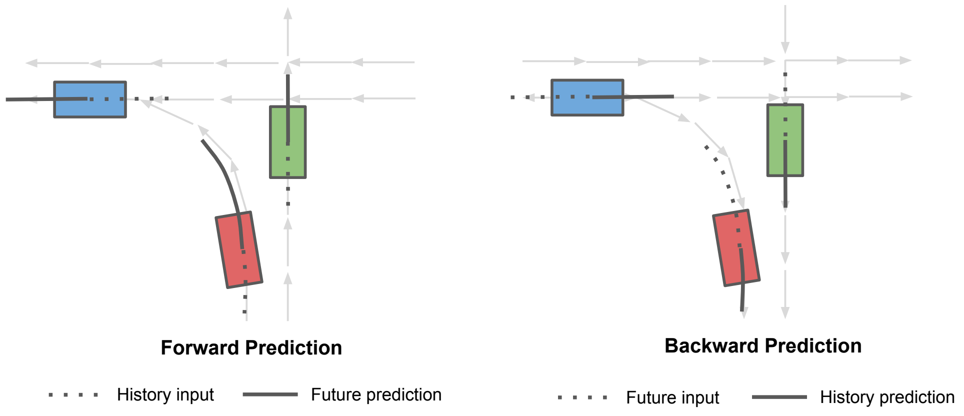

Figure 1: Cycle prediction. In the backward prediction pass, we reverse the trajectories as well as the direction of the road graph, and let the model predict backward in time.

and scale symmetries have been incorporated into the loss function using data augmentation in various methods [9, 14, 15]. [17] incorporates rotation and translation invariance explicitly into the model structure and input representation. [14] enforces temporal and spatial consistency constraints into the loss function, which is shown to improve prediction accuracy.

In this work, we propose a novel consistency-based loss called *cycle loss*, which is illustrated in Figure 1. The key motivation of our method is that an agent's future trajectory should be coherent with its past observations, and this coherency should still hold even if we reverse the sequence backward in time. Motivated by this observation, we propose to add an auxiliary task and loss term in the existing training scheme of the motion forecasting model. After the model predicts the future trajectory using the history input, we reverse the predicted future trajectory backward in time and reverse the direction of the road graph, and we feed them back into the prediction model to let it predict the history. We compute the loss of this auxiliary task as an additional cycle loss term. Similar to the temporal consistency loss proposed in [14], our cycle loss method is generic and can be applied to any motion forecasting model.

We summarize our contributions as follows:

1. We propose *cycle loss*, a novel training scheme and consistency loss for motion forecasting. This loss explicitly ensures that the future trajectories predicted by the model are coherent with the history observations. We also introduce the concept of *ground truth trajectory mixing*, which is necessary for cycle loss to work.
2. We conduct extensive experimentation on the Argoverse dataset [2]. We justify the design choices needed for cycle loss to work through ablations and demonstrate that cycle loss can improve the performance of competitive motion forecasting models.

2 Related Work

Attention-based motion forecasting models: Attention is used widely in motion forecasting models to fuse diverse features, especially agent and lane features. [9] proposed a novel form of graph convolution to aggregate short and long-range lane features and used self and cross attention to fuse agent and lane features. [4, 5, 6] also use self and cross-attention between agent and lane features. More recently, several transformer-based papers such as [10, 7, 11] incorporate temporal attention as well. Some works, such as [17], introduce a novel formulation of attention to enforce spatial and rotational invariance. Other works, such as [15], model the scene using two graph layers and use attention to fuse embeddings of the two layers. In our experiments, we applied our proposed cycle loss method to two attention-based motion forecasting models, GOHOME [5] and Autobots [7]. We selected them because of their popularity and competitive performance.

Consistency-Based Losses: There are several inherent constraints in the motion forecasting problem that have been discussed in the literature. [14] introduced temporal and spatial consistency losses. Temporal consistency enforces that inputs shifted by a small time interval produce similar output trajectories. Spatial consistency enforces that inputs perturbed by a small amount of noise produce similar output trajectories. The loss introduced by [13] is the closest to our work in motion prediction of autonomous vehicles. They pass learning embeddings for the future timesteps through a temporal flow network to reconstruct the input. However, since they use a separate model for reconstruction, it is not truly a consistency loss; but rather a method to enrich the learned embeddings. In our work, we use the same model to predict forward and backward in time, making our method a consistency loss on the model. Another difference between our cycle loss method and [13] is that we use the predicted trajectories to do backward prediction, while [13] uses feature embeddings. [12] uses a similar approach to ours for human trajectory prediction. They pretrain two models for forward and backward prediction and refine the predictions for both models by jointly training both models with cycle loss. This makes their training procedure lengthy as they have to train three times. Our approach differs since we use the same model for forward and backward prediction making our training end-to-end and therefore much faster than [12]. This also makes cycle loss a true consistency loss on the model which is not the case for [12]. The element of *ground truth trajectory mixing* is crucial for our method to work.

3 Methods

3.1 Problem Statement

The general motion forecasting problem involves predicting the future trajectories of agents given the history trajectories and map context (usually represented as a lane graph). Consider that the number of agents in the scene is A, of which P are target agents whose trajectories need to be predicted by the model, and B are background agents which are used to provide scene context for the predictions of the target agents. Let the history length be H and the future length to be predicted be F. We are given the history trajectory of the target agent i as $\mathbf{x}^i = \{x_1^i, \dots, x_H^i\}$ and the history trajectory of background agent i as $\mathbf{b}^i = \{b_1^i, \dots, b_H^i\}$ where each element is a 2D array with x and y coordinates. The map context $\mathcal{M}$ is represented as a directional lane graph, where each node in the graph is a lane segment with a start point and an end point. We denote the ground-truth future positions of target agent i as $\mathbf{y}^i = \{y_1^i, \dots, y_F^i\}$ and background agent i as $\mathbf{b}_f^i = \{y_{f1}^i, \dots, y_{fF}^i\}$. Our task is to predict K future trajectories for each target agent, where the k-th future trajectory is denoted as $\mathbf{y}^{pik} = \{y_1^{pik}, \dots, y_F^{pik}\}$.

3.2 Cycle Consistency Training

The overall training scheme with cycle loss is illustrated in Figure 2. We conduct two passes of the model, a normal *forward prediction pass* that is the same as the regular motion forecasting training and a *backward prediction pass* that reverses the trajectories as well as the lane graph backward in time.

In the *forward prediction pass*, we use the input agent histories, **x** and **b**, along with the map context $\mathcal{M}$, to predict the target agent future trajectories $\mathbf{y}^{pk}$. We use the standard winner-takes-all multimodal prediction loss [3] as the forward loss, which has a mode classification term and a trajectory regression term. The trajectory regression term is computed on the prediction trajectory that is closest to the ground-truth measured by the final displacement error (FDE).

In the *backward prediction pass*, we reverse the future trajectories of the agents as well as the directions of the road graph, and feed them back to the model to predict the history trajectory. When the model predicts $K > 1$ future trajectories for the target agent, we pick the trajectory that is closest to the ground-truth as measured by the final displacement error (FDE), denoted as $\mathbf{y}^p$. And we denote the reversed future trajectory as $\mathbf{y}^{rp}$ where $\mathbf{y}^{rpi} = \{y_F^{pi}, \dots, y_1^{pi}\}$, which is the backward pass input for the target agent. For the background agents that the model doesn't predict for, we use their ground-truth future trajectories to reverse and get $\mathbf{b}_f^r$ where $\mathbf{b}_f^{ri} = \{y_{fF}^i, \dots, y_{f1}^i\}$. We also reverse the map context to obtain $\mathcal{M}^r$. For the map context, we reverse the connection directions of the lane graph, and we reverse the directions of all the lane segments in the graph.

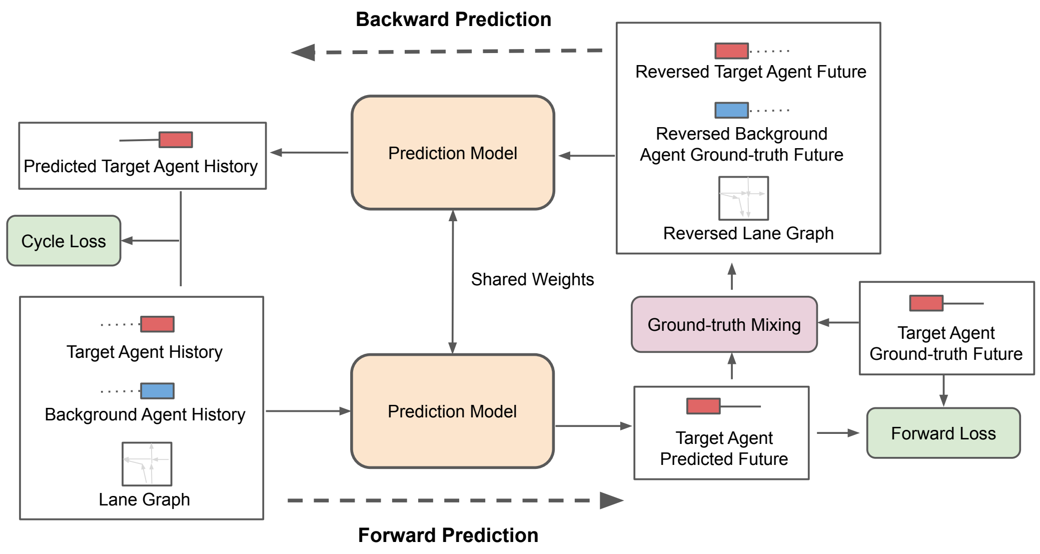

Figure 2: Cycle consistency training architecture

In the simpler case when $F = H$, we use $\mathbf{x}_{rev} = \mathbf{y}^{rp}$ and $\mathbf{b}_{rev} = \mathbf{b}^{r}_{f}$, along with the reversed map context $\mathcal{M}^{r}$, to predict backward in time the target agent history trajectories $\mathbf{y}^{pk}_{rev}$. The predicted history trajectory and the original agent history $\mathbf{x}$ are then used to compute our *cycle loss*. Similar to the forward pass, we use the winner-takes-all approach to compute the cycle loss. We pick the history trajectory that is closest to the original history as measured by the final displacement error (FDE) to calculate the loss.

$$Cycle\ Loss \ = \frac{1}{HP} \sum_{i=1}^{i=P} \min_{k \in \{1, \dots K\}} \sum_{j=1}^{j=H} ||y^{pik}_{rev, H-j} - x^{i}_{j}||_2$$

3.3 Trajectory Truncation or Extension for $F \neq H$

In many datasets (such as Argoverse), the length of the future prediction trajectory F is longer than the history length H, that is, $F > H$. For those datasets, we need to truncate the future trajectory to length H when passing as the input to the backward pass. We provide ablations for different choices of truncation parameters in Section 4.4.1.

Similarly, in datasets where $F < H$, we need to extend the future trajectory with a part of the agent history to make it length H.

3.4 Ground Truth Trajectory Mixing

If we use purely the predicted target agent trajectories as the input to the backward pass, there is a possibility that this encourages the model to predict future trajectories that are easier to regress backward but are less accurate. For example, the model might predict purely straight line modes instead of accounting for right or left turns since straight line modes have only velocity uncertainty as opposed to other modes that have maneuver uncertainty as well.

One straightforward way to mitigate this issue is to reduce the weight of the cycle loss term, but this will limit the improvement offered by cycle loss. In this work, we find a better approach to addressing this issue is to mix the predicted future trajectories with the ground-truth future trajectories with a probability p to generate a *mixed trajectory*, and use the mixed trajectory as input to the backward pass. This ensures that if the predicted future trajectory is too far from the ground truth, the mixed

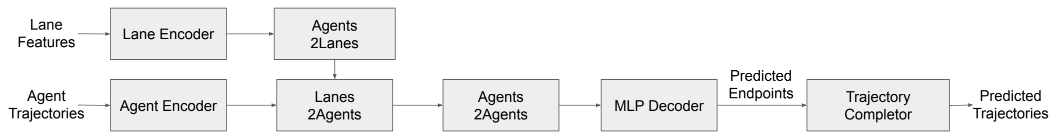

Figure 3: Block diagram for our GOHOME-Mod model

trajectory will be completely infeasible and will result in poor backward predictions and a high cycle loss.

Mathematically, as we described in Section 3.2, after reversing the predicted target agent future trajectories, we obtain $\mathbf{y}^{rp} \in \mathcal{R}^{P\times F\times 2}$. We also flip the ground truth target agent future trajectories to obtain $\mathbf{y}^{r}$ where $\mathbf{y}^{ri} = \{y^i_F, \dots, y^i_1\}$. Here, $\mathbf{y}^{r} \in \mathcal{R}^{P\times F\times 2}$. We generate a binary mask which we denote as $GTM \in \mathcal{R}^{P\times F\times 2}$ where each element of GTM is generated independently and is 1 with probability p and 0 with probability $1-p$. Thus, with ground truth trajectory mixing,

$$\mathbf{x}_{rev} = GTM \otimes \mathbf{y}^{rp} + (1 - GTM) \otimes \mathbf{y}^{r}$$

3.5 Implementation Details

3.5.1 GOHOME Implementation

We implemented our cycle consistency training scheme on GOHOME [5], which is a goal-based motion forecasting model using attentions. It first predicts the endpoint of the trajectory and then completes the whole trajectory conditioned on the goal. Since the original GOHOME model code is not open-sourced, we implemented our own variant of the GOHOME model, and we refer to it as *GOHOME-Mod* for the rest of the paper.

The architecture of our GOHOME-Mod model is illustrated in Figure 3. We use 1D CNN + GRU as our encoder in both agent and lane encoders to encode agent trajectories and lane centerlines. We use cross-attention block *Agents2Lanes* to generate agent-aware lane features and use cross-attention block *Lanes2Agents* to generate lane-aware agent features. We then apply self-attention block *Agents2Agents* to produce inter-aware agent features and use a 3-layer MLP decoder to predict the trajectory endpoints of the target agents. Finally, we use a trajectory regressor module to complete the trajectory conditioned on the predicted endpoint and the agent feature embeddings. Similar to the other goal-based prediction approaches such as [4, 5, 6, 14], we train the trajectory regressor conditioned on the ground truth trajectory endpoints. For all attention blocks, we use multi-head attention with 8 attention heads.

We add our proposed cycle loss as an additional loss term in addition to its original forward losses of GOHOME-Mod. We refer to the model with cycle consistency loss as *GOHOME-Mod + CL*.

3.5.2 Autobots Implementation

We also applied our cycle loss method to the official open-source implementation of Autobots [7], which is a transformer-based motion prediction model. We used the single agent prediction version of the model (Autobots-Ego) since it has better performance on the Argoverse dataset. Same as the GOHOME-Mod + CL model, we added the cycle loss as an additional loss term in addition to its original forward losses. We refer to the model with cycle consistency loss as *Autobots + CL*.

4 Evaluation

4.1 Datasets

We use the Argoverse 1.1 dataset [2], which is a widely used dataset for motion forecasting to evaluate our method. The Argoverse 1.1 dataset consists of real-world driving scenarios with 205,942 training samples, 39,472 validation samples, and 78,143 test samples. For training and validation samples, the dataset provides a 2 second history and a 3 second future trajectories sampled at 10 Hz. For the test samples, only the 2 second history is provided. The dataset is collected from two cities, Miami and Pittsburgh. In addition to the agent trajectories, the map information is also provided in the form of a lane graph. It represents lanes as lane centerlines, which is a sequence of points, with lane attributes including turn direction, presence of traffic control, and presence of intersections. To construct the reversed lane features required by cycle consistency, we reverse the turn directions and reverse the order of points in each centerline. Argoverse is a single-agent dataset that only requires the models to predict the trajectories for one single target agent in each frame. Therefore, using our terminology described in Section 3.1, for the Argoverse dataset, $H = 20$, $F = 30$ and $P = 1$.

We use the minADE, minFDE, MR, and DAC metrics, which are standard metrics used by the Argoverse leaderboard [2]. minFDEK is the minimum displacement between the last waypoint of the ground truth trajectory and the last predicted waypoint of top K predicted trajectories. minADEK is the minimum average displacement between all waypoints of the ground truth trajectory and all predicted waypoints of top K predicted trajectories. MR-K represents the rate at which minFDEK exceeds 2 meters over the entire dataset. DACK represents the percentage of the top K predicted trajectories that do not leave the drivable area at any point. For minADE, minFDE, and MR, lower numbers are better. For DAC, higher numbers are better.

4.2 Experimental Details

For the GOHOME-Mod model, we train the model for 200 epochs with an initial learning rate of 10^{-3} decayed by 0.5 every 60 epochs.

For the Autobots model, we train the model for 150 epochs with an initial learning rate of 7.5×10^{-4} decayed by 0.5 every 20 epochs.

Unless otherwise specified, for both models, we use the first 20 predicted future timesteps to reverse and pass them into the model in the backward prediction pass, and we use a probability of $p = 0.5$ for ground truth trajectory mixing. We set the weight of cycle loss to 2 for GOHOME-Mod and to 1 for Autobots.

4.3 Argoverse Test Results

Model	**minFDE6**	**minADE6**	**DAC6**	**MR6**
THOMAS [6]	1.4388	0.9423	0.9781	**0.1038**
TNT [16]	1.4457	0.9097	**0.9889**	0.1656
GOHOME [5]	1.4503	0.9425	0.9811	0.1048
GOHOME-Mod (ours)	1.4368	0.9426	0.9787	0.1763
GOHOME-Mod + CL (ours)	**1.3896**	**0.8949**	0.9833	0.1682

Table 1: Results on Argoverse Test Set. The baseline numbers are from the leaderboard.

Table 1 shows the results of our models on the Argoverse test set compared to some standard baselines, including THOMAS [6], TNT [16], and the original GOHOME [5] model. The numbers for the baseline models are directly copied from the leaderboard. We can see that, by adding cycle loss, our GOHOME-Mod + CL model outperforms all metrics of its corresponding baseline GOHOME-Mod and improves minFDE6 by a significant amount. This result demonstrates the effectiveness of our cycle loss method. The result also shows our GOHOME-Mod implementation archives similar performance as the reported numbers of the original GOHOME model on the Argoverse leaderboard, indicating that our re-implementation is a competitive model.

4.4 Ablation Studies

We performed the following ablation studies on the Argoverse validation set.

4.4.1 Importance of Predicted and Ground-Truth Positions

Model	Future input	History target	ValFDE6	ValADE6
GOHOME-Mod	-	-	1.163	0.767
GOHOME-Mod + CL	30-50	1-30	1.146	0.7543
GOHOME-Mod + CL	20-40	1-20	**1.104**	**0.7369**

Table 2: Ablation study for the predicted and matched positions used for cycle loss. A position of a-b indicates the positions from the ath timestep to the bth timestep are used for calculating cycle loss. Trajectories in Argoverse 1 have a total of 50 timesteps (20 history + 30 future).

In Argoverse, the length of the future trajectory is longer than the length of the history ($H = 20$ and $F = 30$). As a result, in the backward prediction pass, we need to pick which part of the future trajectory we feed as the backward input. In this ablation study, we studied different choices of this parameter, as summarized in Table 2.

The result shows that the model performs the best when we feed in the first 20 future waypoints as the backward pass input and use them to predict the 20 history waypoints. The losses for the remaining part of the backward prediction trajectories are masked out. We believe the reason why this setting yields the best performance is because the uncertainty in the last 20 positions is larger compared to the uncertainty in the first 20 positions, making it tough for the model to obey cycle consistency.

4.4.2 Importance of Ground Truth Trajectory Mixing

Model	Mixing probability (p)	ValFDE6
GOHOME-Mod	-	1.163
GOHOME-Mod + CL	0 (All Ground Truth Positions)	1.133
GOHOME-Mod + CL	1 (All Predicted Positions)	1.184
GOHOME-Mod + CL	0.5	**1.104**

Table 3: Ablation study for the mixing probability of ground truth positions and predicted positions. The first 20 predicted waypoints are used to feed into the model, and the 20 input positions are used for calculating cycle loss, as is found optimal in Section 4.4.1

In Table 3, we studied different parameters of ground-truth mixing (described in Section 3.4). As we can see from the results, mixing ground truth future trajectories with the predicted future trajectories gives us the best FDE on the validation set. An interesting fact to note is that, when we use all ground truth futures as the backward prediction inputs, it can also reduce the FDE by an amount of 3 cm over the baseline. Since using all ground truth futures essentially corresponds to augmenting the dataset using the time-reversed data, the result suggests that time-reversing can also be used as an augmentation strategy to enhance the training of the motion forecasting models.

We believe the main reason why *time flipping augmentation* helps in motion forecasting is that it reduces the imbalances of complex maneuvers in the dataset. For example, if the dataset has more left turns than right turns, *time flipping augmentation* can balance the number of those maneuvers. Similarly, a car might increase its speed or decrease its speed in the future trajectory. If the number of instances of the car speeding up are higher than the car slowing down, that creates an imbalance which *time flipping augmentation* can again correct.

Meanwhile, when we use all predicted futures, its performance is worse than not applying cycle loss at all. This highlights the importance of ground truth trajectory mixing, which ensures that the predicted trajectory does not stray off the ground truth in a bid to minimize the cycle loss.

4.5 Additional Results on Autobots

In addition to GOHOME-Mod, we also implemented cycle loss on Autobots [7]. The design choices made were the same as in the GOHOME-Mod model, except that the weight of cycle loss was fixed at 1 based on our parameter search.

Model	minFDE6
Autobots [7]	1.114
Autobots + CL	**1.084**

Table 4: Autobots Results on Argoverse Validation Set

As the results in Table 4 show, the addition of cycle loss improves the prediction performance of the open-sourced Autobots model as well. This result demonstrates that our cycle loss method is generic.

5 Limitations

There are several limitations to our proposed consistency loss method. First, similar to the temporal consistency loss proposed in [14], the cycle loss method incurs overhead at the training time, as it requires two passes of the model during training, nearly doubling the training time. Second, some of the cycle loss parameters depend on the model architecture and dataset. For example, we find the optimal cycle loss weight is different in GOHOME-Mod and in Autobots. Third, cycle loss requires the model to be end-to-end differentiable. Some models [5, 6, 4, 8] rely on sampling a probability distribution as an intermediate step, and more sophisticated designs will be needed in order to apply cycle loss to those models.

6 Conclusion

In this work, we propose cycle loss, a novel consistency-based training scheme and loss for motion forecasting. We demonstrated its effectiveness on two competitive motion forecasting models on the Argoverse dataset. The result shows that cycle loss is able to improve the prediction performance of those models by a significant margin.

References

[1] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 86–99. PMLR, 30 Oct–01 Nov 2020.

[2] Ming-Fang Chang, John W Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] 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.

[4] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. Home: Heatmap output for future motion estimation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 500–507, 2021. doi: 10.1109/ITSC48978.2021.9564944.

[5] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 9107–9114, 2022. doi: 10.1109/ICRA46639.2022.9812253.

[6] Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. THOMAS: Trajectory heatmap output with learned multi-agent sampling. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=QDdJhACYrlX`.

[7] Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=Dup_dDqkZC5`.

[8] Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021.

[9] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *ECCV*, 2020.

[10] Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S. Refaat, and Benjamin Sapp. Wayformer: Motion forecasting via simple amp; efficient attention networks. 2022. doi: 10.48550/ARXIV.2207.05844. URL `https://arxiv.org/abs/2207.05844`.

[11] Jiquan Ngiam, Vijay Vasudevan, Benjamin Caine, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, David J Weiss, Ben Sapp, Zhifeng Chen, and Jonathon Shlens. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=Wm3EA5OlHsG`.

[12] Hao Sun, Zhiqun Zhao, and Zhihai He. Reciprocal learning networks for human trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[13] Yuting Wang, Hangning Zhou, Zhigang Zhang, Chen Feng, Huadong Lin, Chaofei Gao, Yizhi Tang, Zhenting Zhao, Shiyu Zhang, Jie Guo, Xuefeng Wang, Ziyao Xu, and Chi Zhang. Tenet: Transformer encoding network for effective temporal flow on motion prediction. 2022. URL `https://arxiv.org/abs/2207.00170`.

[14] Maosheng Ye, Jiamiao Xu, Xunnong Xu, Tongyi Cao, and Qifeng Chen. Dcms: Motion forecasting with dual consistency and multi-pseudo-target supervision. 2022.

[15] Lu Zhang, Peiliang Li, Jing Chen, and Shaojie Shen. Trajectory prediction with graph-based dual-scale context fusion. *arXiv preprint arXiv:2111.01592*, 2021.

[16] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, Congcong Li, and Dragomir Anguelov. Tnt: Target-driven trajectory prediction. In *CoRL*, 2020.

[17] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8823–8833, June 2022.