2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Pre-training on Synthetic Driving Data for Trajectory Prediction

Yiheng Li[1*] Seth Z. Zhao[1*] Chenfeng Xu[1] Chen Tang[1] Chenran Li[1] Mingyu Ding[1]
Masayoshi Tomizuka[1] Wei Zhan[1]

***Abstract*— Accumulating substantial volumes of real-world driving data proves pivotal in the realm of trajectory forecasting for autonomous driving. Given the heavy reliance of current trajectory forecasting models on data-driven methodologies, we aim to tackle the challenge of learning general trajectory forecasting representations under limited data availability. We propose a pipeline-level solution to mitigate the issue of data scarcity in trajectory forecasting. The solution is composed of two parts: firstly, we adopt HD map augmentation and trajectory synthesis for generating driving data, and then we learn representations by pre-training on them. Specifically, we apply vector transformations to reshape the maps, and then employ a rule-based model to generate trajectories on both original and augmented scenes; thus enlarging the driving data without collecting additional real ones. To foster the learning of general representations within this augmented dataset, we comprehensively explore the different pre-training strategies, including extending the concept of a Masked AutoEncoder (MAE) for trajectory forecasting. Without bells and whistles, our proposed pipeline-level solution is general, simple, yet effective: we conduct extensive experiments to demonstrate the effectiveness of our data expansion and pre-training strategies, which outperform the baseline prediction model by large margins, *e.g.* 5.04%, 3.84% and 8.30% in terms of MR_6, $minADE_6$ and $minFDE_6$. The pre-training dataset and the codes for pre-training and fine-tuning are released at `https://github.com/yhli123/Pretraining_on_Synthetic_Driving_Data_for_Trajectory_Prediction`.**

I. Introduction

Trajectory forecasting is important for safely navigating autonomous vehicles in crowded traffic scenarios. The state-of-the-art trajectory forecasting models are empowered by data-driven supervised learning approaches, whose performance heavily relies on the scale of motion data available for training [1]–[3]. However, driving data is expensive and time-consuming to collect and annotate, which hinders cost-efficient scaling of training data as in Natural Language Processing (NLP) and Computer Vision (CV). Data-collection vehicles with sophisticated sensors need to run on public roads to collect traffic data. For instance, the Argoverse V1.1 dataset [4] and the Waymo Motion Dataset [1] consist of 324K and 103K driving scenes, which add up to a total of 320 and 574 hours of driving data, respectively. Notably, a substantially larger amount of raw data needs to be collected to filter out the given amount of high-quality training data. In addition to data collection, a significant amount of human labor is required to annotate the raw data into a cohesive dataset—traffic participants need to be annotated, and HD maps need to be aligned and integrated with the traffic data.

TABLE I
Effortless Driving Data Generation.

Dataset	Driving Hours	Scenes
Argoverse v1.1	320	324k
Synthetic Dataset	+0	+370k

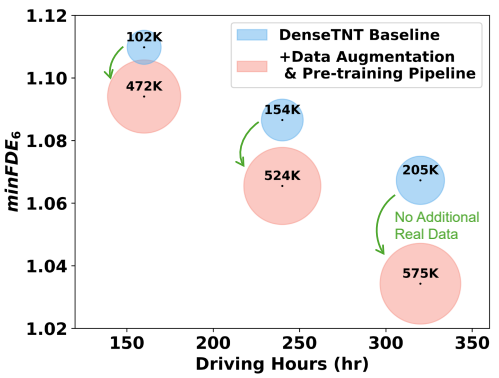

Fig. 1. **Our data synthesis and self-supervised pre-training pipeline enhances prediction performance without extra real-world driving data.** Each circle's area is proportional to the total number of synthetic and real driving scenes used, as indicated within. (A lower $minFDE_6$ is preferable.)

In this work, we sought to *synthetic data* as a solution to break through the data bottleneck in trajectory forecasting. Compared to collecting and processing real-world data, which involves huge human labor, automatically generating synthetic data is effortless. As shown in Tab. I, our proposed data generation method only takes 20 *computational hours* to generate a comparable number of scenes to the real-world dataset, which requires hundreds of *human driving hours*. More importantly, as shown in Fig. 1, the synthetic data can significantly improve the prediction accuracy and data efficiency when paired with our proposed training pipeline.

Concretely, we propose a pipeline-level solution with two parts: we generate the synthetic data and then learn representation by pre-training on them. The generation part consists of a map augmentation and a trajectory generation process. For map augmentation, we adopt the vector-transformation method [5] to convert the linear lanes found in real-world maps into curved lanes within a defined range of sharpness and angle, as shown in Fig. 2 (upper-left). The augmented curved roads are introduced to diversify the map data. Subsequently, we navigate a simulated vehicle on both real-world and augmented maps with a rule-based planner [6] to generate trajectory data. The rule-based planner leverages

*Equal contribution

[1] Y. Li, S. Z. Zhao, C. Xu, C. Tang, C. Li, M. Ding, M. Tomizuka, and W. Zhan are affiliated with the University of California, Berkeley. {yhli, sethzhao506, xuchenfeng, chen_tang, chenran_li, myding, tomizuka, wzhan}@berkeley.edu. Correspondence to: Chen Tang.

prior knowledge to facilitate realistic generated trajectories.

Leveraging the generated data is an open question in robotics, NLP, and computer vision [7]–[10]. We conduct an extensive study on various training algorithms that use synthetic data to learn general representations for trajectory forecasting. In particular, we compare three different paradigms: augmenting the real-world prediction dataset [11]–[13], supervised pre-training [14]–[16], and self-supervised pre-training [17]–[27]. The goal is to train a generalizable scene representation on the synthetic data that oversees detailed domain-specific information and can benefit cross-domain fine-tuning. Extensive experiments on the Argoverse dataset with denseTNT backbone demonstrate that self-supervised pre-training outperforms the other two methods. With our synthetic data generation and self-supervised pre-training pipeline, the fine-tuned model outperforms the baseline model by a large margin—5.04%, 3.84%, and 8.30% in terms of MR_6, $minADE_6$ and $minFDE_6$. Detailed ablation studies are then conducted to shed light on the influence of some subtle factors on prediction performance, such as information leaks and map augmentation.

We summarize our contributions below:

- We propose a simple yet effective pipeline for learning general representations in trajectory forecasting at the low-data regime. Our pipeline highlights the synergy of the synthetic map and trajectory generation and representation learning via pre-training on the synthetic data. This pipeline does not introduce any extra human effort while drastically improving the volume of data for the training and enhancing the representation of learning under the scarcity of real driving data.
- We conduct extensive experiments to pinpoint the optimal training scheme to utilize the synthetic data. Our results show that self-supervised pre-training performs better than the other widely explored alternatives, such as directly augmenting the training dataset and supervised pre-training.
- With our synthetic data generation and self-supervised pre-training pipeline, the fine-tuned model outperforms the baseline model by a large margin—5.04%, 3.84%, and 8.30% in terms of MR_6, $minADE_6$ and $minFDE_6$.

II. Related Works

A. Data Augmentation in Trajectory Prediction

Data scarcity is a bottleneck for exploring the model's capacity for better performance [3]. To address this issue, numerous data augmentation techniques, such as the application of image transformations [19], [28], [29] and utilization of synthetic data [29]–[32], are introduced in the CV community to augment or replicate seldom-seen real-world scenes. In the trajectory prediction problem, the unique properties of trajectory and map data necessitate a specific design for data augmentation or generation that is both reasonable and cost-effective. In [33]'s approach, a simple augmentation technique is applied to the vehicle's velocity properties to mimic unpredictable driver behaviors in anticipation of future trajectories based on past trajectories. Additionally, [5] proposes using map augmentation techniques to alter the topological semantics of map information as adversarial attacks on current trajectory prediction models. While these methods expand the dataset size, their additional trajectories are simple geometric transformations of existing lane elements or trajectories without carefully considering the transformed trajectories' feasibility and realism. In our work, we enhance the map data to supplement limited features, like curves and turns, and employ a rule-based simulation to mimic reasonable traffic behaviors for trajectory generation.

B. Pre-training in Trajectory Prediction

Pre-training is effective for the data shortage problem in trajectory prediction. In [3], contrastive learning was used to cluster similar map patches and establish associations between map patches and the trajectory, thereby fully utilizing the existing map data. In [34], four pre-training tasks were initiated to enhance the encoding, including randomly masking and recovering parts of the features. Concurrently, [35], [36] used MAE to pre-train the encoder. However, these methods did not introduce any new data. Another work [33] generated a pseudo trajectory that strictly follows the lanes for pre-training. However, this strict compliance hinders the realism of the pseudo trajectories. In contrast, our work aims to generate realistic synthetic trajectories to mitigate the domain gap between the synthetic and real data.

While limited efforts on pre-training exist in the trajectory forecasting literature, some auxiliary tasks have been introduced to regularize trajectory prediction models during training. [37] introduced a feature-masking task to encourage feature interactions, which inspired us to use methods in CV [21] and NLP [22] to stimulate feature interactions in the learned scene representation. Taking cues from these works, we investigated masked reconstruction as a self-supervised pre-training method to utilize the synthetic data.

III. A Pipeline-Level Solution of Learning General Representations

A. Data Generation

Our data synthesis method is simple yet enables expanding the diversity of trajectory data. The key components include a map augmentation module and a model-based planning model to generate trajectories.

1) Map Data Augmentation: Considering the lack of curve trajectory in the real-world training dataset, we add the map augmentation module to our data synthesis pipeline. Inspired by the conditional adversarial scene generation process proposed in [5], we design transformation functions that alter the original map topology, as shown in Fig. 2 (upper-left). Given that each scene is composed of a set of scene points, we define the transformation on each scene point (s_x, s_y) in the following form:

$$\bar{s} = (s_x, s_y + f(s_x - b)) \tag{1}$$

where $\bar{s}$ is the transformed point, f is the single-variable transformation function, and b is a parameter that determines

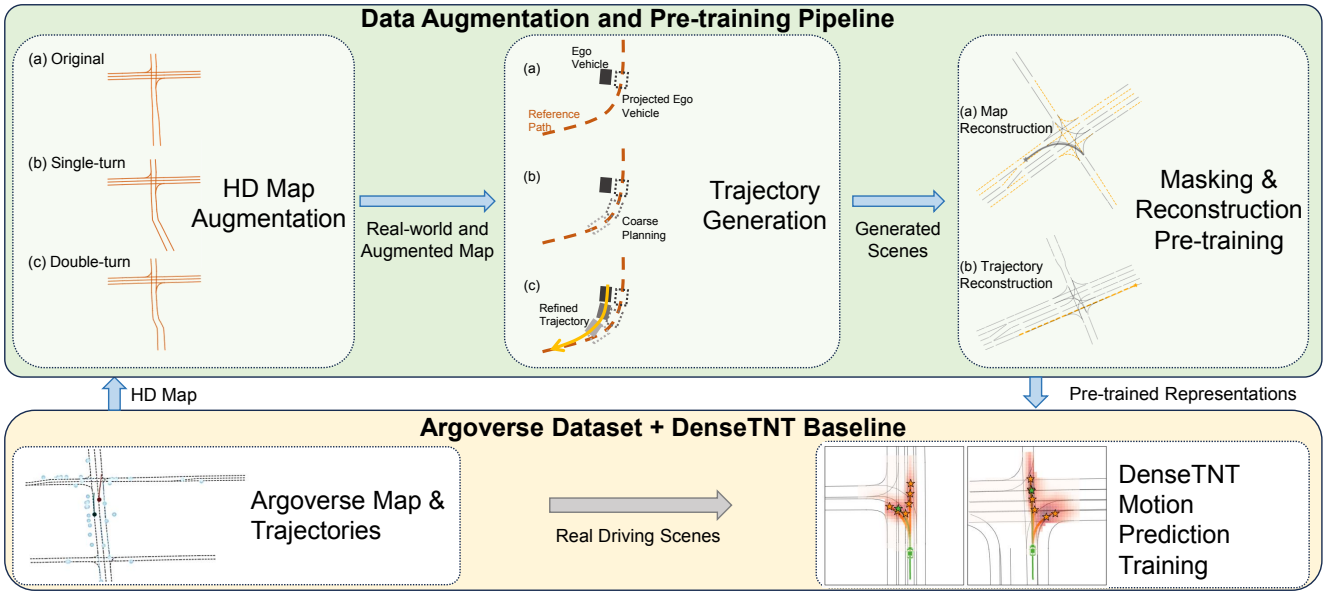

Fig. 2. **Pipeline of driving data synthesis and utilization.** We augment the map and generate trajectory on it to acquire synthetic motion data, which are then used for pre-training via masking and reconstruction. The pre-trained model is used to initialize the backbone model for fine-tuning.

the region of applying the transformation, i.e., we only modify the scene points whose x-coordinates surpass b. We further adopt the following two transformation functions for our map augmentation process.

Single-turn introduces a single turning on the roadway. The transformation function for single-turn is defined as:

$$f_{\text{single-turn}}(s_x) = \begin{cases} 0, & s_x < 0, \\ q_\alpha(s_x), & 0 \le s_x \le s_t \\ (s_x - s_t)q'_\alpha(s_t) + q_\alpha(s_t), & s_t < s_x \end{cases} \tag{2}$$

where s_t represents the length of the turn and q_α is an auxiliary function defined by

$$q_\alpha(s_x) = \frac{\alpha_1}{s_t^{\alpha_2}} s_x^{\alpha_2} \tag{3}$$

where α_1, α_2 are the parameters to control the turn's sharpness and angle. Note the q'_α is simply the derivative of q_α. Our formulation of $f_{\text{single-turn}}(s_x)$ is continuously differentiable and thus makes a smooth augmented single-turn curve.

Double-turn introduces two consecutive smooth single-turns in opposite directions to the road. The transformation function for double-turn is based on the that of single-turn:

$$f_{\text{double-turn}}(s_x) = f_{\text{single-turn}}(s_x) - f_{\text{single-turn}}(s_x - \beta) \tag{4}$$

where β is the distance between two turns. In practice, we randomly select single-turn or double-turn augmented lanes and randomly assign values in a specific range to ensure a reasonable extent of transformation, as shown in Tab. II.

2) Trajectory Data Generation: To generate labeled trajectory data on both real-world and augmented maps, we implement a trajectory data generator to synthesize pseudo-expert trajectories in single-car scenarios, as illustrated in Fig. 2 (upper-middle). Following the motion data generation pipeline proposed in [6], we project its position onto the nearest reference path obtained from the training dataset to determine the optimal trajectory for the ego vehicle. Subsequently, we utilize the A* planning method on this reference path to generate a preliminary, coarse trajectory. Specifically, we define a node n_i in the search space by a tuple (s_i, v_i, t_i), which respectively represents the ego vehicle's coordinate, velocity, and time on the reference path. A transition from n_i to n_{i+1} defines the n_{i+1} by:

$$(s_{i+1}, v_{i+1}, t_{i+1}) = (s_i + v_i\delta t + \frac{1}{2}a\delta t^2, v_i + a\delta t, t_i + \delta t) \tag{5}$$

where a denotes the acceleration and δt denotes the minimum time interval for coarse planning. The cost function of the transition is given as:

$$\mathscr{C}(n_i, a, n_{i+1}) = w_1 a^2 + w_2 \kappa(s_{i+1}) v_{i+1}^2 + w_3 (v_{i+1} - v_d)^2 \tag{6}$$

where w_1, w_2, w_3 are weights for each term, $\kappa(s_{i+1})$ denotes curvature of reference path at s_{i+1}, and v_d denotes desired driving velocity set in prior. The planning process concludes when the time associated with the searched node surpasses the end time t_g, resulting in the conversion of the resultant path into a coarse trajectory $s_c(t)$ within the global frame. The hyperparameters adopted for the synthetic trajectory generation pipeline are listed in Table II. Finally,

TABLE II

PARAMETER SELECTION FOR DATA SYNTHESIS.

Traj. Param.	Value	Map Param.	Value
a	$\in [-2, 1]$	s_x	10
w_1	5	α_1	$\in [1, 10]$
w_2	5	α_2	20
w_3	1	s_t	10
v_d	$\in [6, 15]$	β	20

we introduce an additional refinement procedure to map the final pseudo-expert trajectory's initial state to the ego car's current state. Concretely, we solve the following optimization problem to acquire the refined trajectory $s_r(t)$:

$$\min_{s_r} \sum_{T=\delta\hat{t}}^{t_g} \omega_1 a_r(T)^2 + \omega_2 j_r(T)^2 + \sum_{T=k\delta\hat{t}}^{t_g} \omega_3 (s_r(T) - s_c(T))^2$$
$$\text{s.t.} \quad v_r(T_0) = v_0 \quad \text{and} \quad s_r(T_0) = s_0 \tag{7}$$

where $\omega_1, \omega_2, \omega_3$ are weights for each term, a_r, j_r are refined acceleration and jerk, $\delta\hat{t}$ is the fine-grained time step such that $k\delta\hat{t} = \delta t$. v_0, s_0 corresponds to the ego car's initial velocity and position.

B. Representation Learning on Synthesis Data

Representation learning on synthesis data is widely explored in other fields [7]–[9]. As far as we know, we are the first to make use of generated data in trajectory forecasting for autonomous driving. We comprehensively study several commonly used approaches to identify the best strategy. The first approach is to directly combine the synthetic and real-world data for training, i.e., treating them identically. While this method is straightforward, it introduces bias into the model due to the domain gap between the synthetic and real-world data. Another method is supervised pre-training: we first pre-train the model for the supervised trajectory prediction task on the synthetic dataset and then fine-tune the model on the real-world dataset. While the pre-training and fine-tuning tasks are both supervised prediction tasks, the two-stage scheme is introduced to mitigate the bias caused by the domain gap. Then, we investigate the self-supervised pre-training method elaborated as follows:

1) Self-supervised Pre-training Framework: We adapt MAE [21] into a self-supervised pre-training framework for trajectory forecasting. The goal is to learn a generalizable scene representation that can capture the interactions between lanes and the ones between lanes and trajectories from the synthetic data well. During the pre-training stage, a specific percentage of trajectories or map lanes are masked, challenging the model to reconstruct the masked segments. The masked objects are replaced with mask tokens, which are learned and shared parameters that signal the existence of a masked object to be reconstructed [21]. Positional encoding is added to provide the subordinate and sequential information. Though specific information is removed in mask tokens, they still provide the model with where the masked information is. Such information leak will lead the model to find a shortcut to interpolate missing information instead of reconstructing them by context understanding [38]. Only the unmasked elements are processed through the backbone's encoder to prevent this. After that, the masked tokens, in combination with the encoded unmasked parts, are fed into a shallow transformer decoder to reconstruct the masked parts. Upon completion of the pre-training phase, the backbone model is initialized using the encoder parameters of the pre-trained model and then fine-tuned on real-world data for the prediction task. This approach enables the model to acquire general semantics during pre-training while discarding extraneous details.

2) Masking Strategy: As each lane or trajectory is not as strongly correlated to other lanes or trajectories as the pixels in a figure, we need to design masking strategies to apply MAE to the trajectory prediction problem. Specifically, we propose and compare different design options for masking and reconstruction during the pre-training stage:

Map reconstruction is proposed to foster the encoder to capture the lane features and their interactions. The masking policies and reconstruction procedures are illustrated in Fig. 2 (upper-right). We randomly choose 50% lanes as masking lanes, replacing everything but the coordinate information for their first points with mask tokens. These masked objects, together with the encoded features of the unmasked objects, are then fed into the decoder. Utilizing lane-wise interactions, the feature of the masked lanes would be recovered. Point-wise L1 loss is employed across all our experiments. The loss function of map reconstruction l_{map} is the difference between the reconstructed lane and the ground-truth one.

Trajectory reconstruction is introduced to improve the model's understanding of the trajectory features and the interactions between the trajectory and map elements, as shown in Fig. 2 (upper-right). Since the synthetic scene only consists of one agent, only a single trajectory is masked in each scene. We retain its starting point in the masked object as in map reconstruction. To prevent mode collapse, the decoder is allowed to produce six potential trajectories. We select the one with the minimum loss to define the loss function l_{traj}. Additionally, to ensure that all modes are activated, we add the weighted reconstruction errors of the remaining modes to the loss function as regularization. A weight of 0.05 is used in our experiments.

Combination of Map and Trajectory Reconstructions integrates both map and trajectory reconstruction tasks by randomly selecting a percentage of scenes within a mini-batch to conduct either of the reconstructions. This selection is achieved by generating a random number prior to masking each scene. As a result, the same scene could undergo different reconstruction tasks across various epochs. The overall loss function is computed as the average of the individual scene-specific losses previously described.

IV. Experiments

A. Synthetic Dataset

Our synthetic dataset is generated in accordance with the methods outlined in Sec. III-A. The map is sourced from the HD maps in the Argoverse v1.1 Motion Forecasting Dataset [4]. We generate trajectories on both the original map and the augmented map. The resultant generated dataset contains 370k driving scenes, where 205k are generated on the original map and 165k are generated on the augmented map. Each generated scene contains the vehicle's position spanning 5 seconds, following the format of [4]. As visualized in Fig. 3, the generated data align well with a vast spectrum of velocities in real-world data distribution. While our synthetic dataset encompasses a broader spectrum of

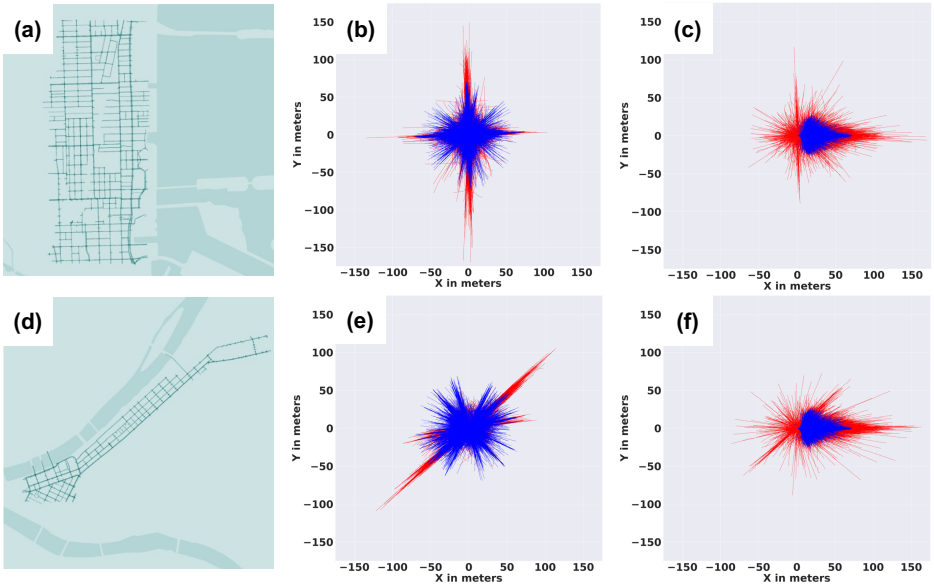

Fig. 3. **Data distribution comparison between synthetic dataset (blue) and the real-world dataset (red) in terms of the speed and direction properties.** (a) represents the map representation for MIA city. (b) represents the trajectory distribution of scenes in (a), showing a pattern of divergence in velocity properties. (c) represents the trajectory distribution of scenes in (b) rotated to the same initial direction, demonstrating a pattern of divergence in direction properties. (d)(e)(f) are the counterparts of (a)(b)(c) in PIT city.

velocity than the real-world data, it is still important to note that vehicles may still drive under extreme velocity in some long-tailed events present in the real-world data. Additionally, the real-world dataset encompasses a multitude of scenarios with pronounced turning angles, such as U-turns, which pose challenges in accurate generation within the synthetic dataset. The discrepancy in data distributions validates that self-supervised pre-training is necessary to mitigate the impact of distributional shifts.

B. Prediction Experiment Setup

Argoverse v1.1 Motion Forecasting Dataset. The Argoverse v1.1 Motion Forecasting Dataset [4] is a renowned dataset frequently employed in vehicle motion prediction studies. It consists of 324k scenes, each spanning 5 seconds and sampled at 10 Hz. In every scene, the initial 2 seconds serve as the history, while the subsequent 3 seconds are designated for prediction. The dataset also includes HD maps collected from Pittsburgh and Miami, covering all the scenes in the dataset. There are 205k and 39k scenes in the training and validation sets, respectively, in which the validation set is used for evaluating our model.

Evaluation Metrics. Following Argoverse's benchmark, we use miss rate (MR_6), minimum final displacement error ($minFDE_6$), and minimum average displacement error ($minADE_6$) as the evaluation metrics. The subscripts mean that the minima are computed over six predicted trajectories.

Implementation Details. In our experiments, we adopt DenseTNT [39] as the prediction backbone. DenseTNT uses Vectornet [37] as its encoder, which adopts a vectorized scene representation, i.e., the lanes and trajectories are represented as vectors, which makes the masking process straightforward. The pre-training is conducted on an NVIDIA A6000 GPU with a batch size of 256. The learning rate is 10^{-5} for trajectory reconstruction and 10^{-3} for map reconstruction. We use a learning rate of 10^{-4} for mixed map and trajectory reconstructions. Except for the GPU, all other fine-tuning settings are the same as in DenseTNT [39]. During evaluation, we choose the 100ms optimization for $minFDE_6$.

TABLE III

PERFORMANCE OF SELF-SUPERVISED PRE-TRAINING.

Method	MR_6(%) (↓)	$minFDE_6$ (↓)	$minADE_6$ (↓)
Baseline	9.73	1.0673	0.8052
Map Reconstruction	**9.20 (-5.45%)**	1.0343 (-3.09%)	0.7571 (-5.97%)
Trajectory Reconstruction	9.24 (-5.04%)	**1.0263 (-3.84%)**	0.7384 (-8.30%)
Combined Reconstruction	9.27 (-4.73%)	1.0349 (-3.04%)	**0.7284 (-9.54%)**

C. Performance of the Proposed Pipeline

We present the trajectory prediction performances of our pipeline with different self-supervised pre-training tasks in Tab. III. The choice of adopting self-supervised pre-training for generated data results from thorough performance comparisons of different representation learning methods in Part V-C. The experiment results show that all of the self-supervised pre-training tasks can substantially improve the prediction accuracy compared with the baseline. Specifically, with map reconstruction, an MR_6 of 9.20% is observed, which is 5.04% lower than the baseline. Trajectory reconstruction pre-training helps improve the $minFDE_6$ by 3.84% compared to the baseline. Combining them together, when we allocate 70% scenes in a batch to map reconstruction and 30% scenes to trajectory reconstruction, we achieve a $minADE_6$ of 0.7284, marking an improvement of 9.54% against the baseline.

To elucidate the impact of our synthetic data and the pre-training, we provide qualitative examples in Fig. 4.

TABLE IV

CONTRIBUTIONS OF SYNTHETIC DATA.

Pretrain Dataset	$MR_6(\%)$	$minFDE_6$	$minADE_6$
Baseline	9.73	1.0673	0.8052
Argoverse Dataset	9.68	1.0517	0.7571
Synthetic Dataset	**9.24**	**1.0263**	**0.7384**

TABLE V

CONTRIBUTIONS OF MAP AUGMENTATION.

Pretrain Dataset	$MR_6(\%)$	$minFDE_6$	$minADE_6$
Baseline	9.73	1.0673	0.8052
Original Map	9.40	1.0517	0.7694
Original+Augmented Map	**9.20**	**1.0343**	**0.7571**

TABLE VI

METHOD COMPARISON FOR USING SYNTHETIC DATA.

PT=PRE-TRAINING

Method	Pretrain Task	$MR_6(\%)$ (↓)	$minFDE_6$ (↓)	$minADE_6$ (↓)
Baseline	/	9.73	1.0673	0.8052
Dataset Augmentation	/	9.45	1.0567	0.7432
Supervised whole PT	Prediction	9.34	1.0371	0.7551
Supervised encoder PT	Prediction	**9.20**	1.0369	0.7639
Self-supervised PT (Ours)	Reconstruction	9.24	**1.0263**	**0.7384**

Fig. 4 (a) and (b) contrast prediction outcomes without and with trajectory reconstruction pre-training. We observe that the model without pre-training predicts potential right-turn trajectories. However, since the target vehicle is not in the right-turn lane, its intention to go straight should be able to be identified, if the encoder captures the spatial relation between the historical trajectory and the lanes. In contrast, trajectory pre-training strengthens the model's confidence in discerning the go-straight intention. Sub-figures (c) and (d) compare the prediction outcomes of the models without and with map reconstruction pre-training. In these sub-figures, the right-turn lane eventually merges with the 2nd lane from the right. We observe that with map pre-training, the model can more accurately predict that the vehicle will not merge into the 1st lane from the right. This result aligns with expectations—map reconstruction enhances the model's understanding of lane connectivity, thereby improving prediction accuracy.

V. DISCUSSION

In this section, we elaborate on some detailed design choices we made. In Sec V-A and Sec. V-B, we validate the necessity of driving data synthesis and map augmentation. In Sec V-C, we compare different representation learning methods for utilizing generated data. In Sec. V-D, we look into the information leak problem and validate our pre-training model architecture choice, which avoids information leaks and effectively improves the fine-tuned model's performance.

A. Benefits of Synthetic Driving Data

A straightforward baseline is directly performing pre-training on the given real dataset. To demonstrate the superiority of using the effortlessly obtained synthetic data, we compare the performance of the trajectory forecasting model pre-trained on the synthetic and real-world data, respectively, as shown in Tab. IV. We observe that pre-training on the real-world dataset does not bring as much improvement as utilizing our synthetic data. Specifically, pre-training on the synthetic data improves that on the real data over 0.44% points, 0.0254, and 0.0187 points in terms of MR_6, $minFDE_6$, and $minADE_6$, respectively, which emphasizes the benefits of our method in both its effectiveness and its efficiency in exploiting real-world driving data.

B. Benefits of Map Augmentation

Previous work [3] shows that directly pre-training for learning representations regarding the HD map also matters, and recent work [40] also indicates the importance of adapting trajectory prediction methods to diverse maps. Therefore, following [3], we assess the performance of models pre-trained with and without scenes generated on the augmented maps. We also utilize map reconstruction as the proxy task for pre-training. The comparison experiments are shown in Tab. V. It can be seen that the model trained on data with augmented maps significantly outperforms the other. This underscores the advantage of map augmentation in improving prediction models.

C. Comparison of Different Representation Learning Methods

We compare the performances of multiple methods described in Sec. III-B. The results are shown in Tab. VI. While directly augmenting the training dataset can slightly improve the prediction accuracy, it does not perform as well as pre-training approaches. The result is consistent with our motivation behind adopting pre-training methods. We compared two alternative fine-tuning strategies for supervised pre-training methods: initializing the entire model with the pre-trained model or only initializing the *encoder* with the pre-trained one. We found that the decoder does not benefit from loading the parameters from the pre-trained model. We hypothesize that the domain-specific nuances learned by the decoder during supervised pre-training might not generalize well to real-world scenarios. This implies that synthetic data is most suitable for specifically improving representation learning rather than directly improving the entire backbone model for trajectory prediction. To this end, self-supervised pre-training should be the most efficient way to utilize the synthetic data, as the training tasks are deliberately designed for learning generalizable scene representation, which is supported by the experimental results—The self-supervised pre-trained model achieves the best prediction accuracy after fine-tuning; therefore it is chosen in our pipeline.

D. Information Leak in Self-supervised Pre-training

In our unsupervised pre-training process, we adopt a method similar to MAE [21], where the mask objects, including the unmasked first point, are *not* passed to the encoder. While this eliminates the information leak from mask tokens, the encoder also loses the first point coordinates

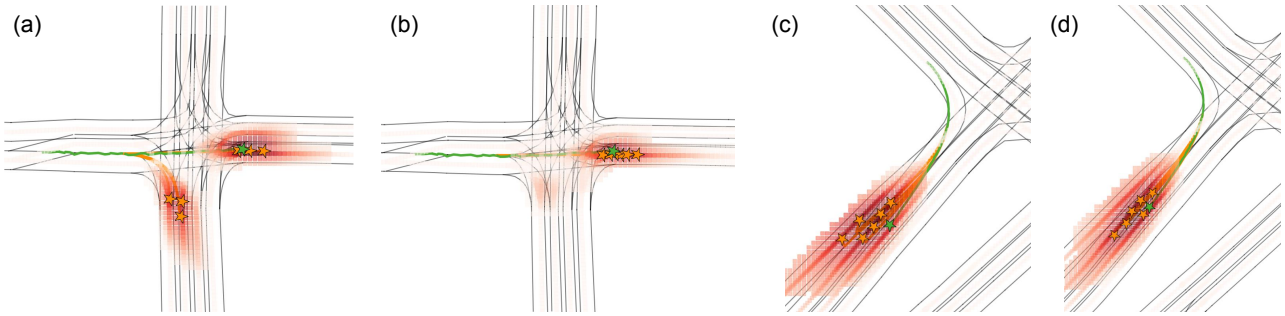

Fig. 4. **Performance comparison without or with pre-training.** The gray lines indicate lane boundaries. The green line and star indicate the true trajectory and its last point, while orange ones are the predicted ones. The orange background shows the possibility of each point being the predicted last point of the trajectory. (a) and (b) show the prediction results without or with trajectory pre-training. (c) and (d) illustrate the performance without or with map pre-training.

TABLE VII
INFLUENCE OF INFORMATION LEAK. PT=PRE-TRAINING.

Pretrain Method	$MR_6(\%)$	$minFDE_6$	$minADE_6$
Baseline	9.73	1.0673	0.8052
No Info Leak Traj PT	9.24	**1.0263**	**0.7384**
W/ Info Leak Traj PT	9.45	1.0520	0.7581
No Info Leak Map PT	**9.20**	1.0343	0.7571
W/ Info Leak Map PT	9.84	1.0735	0.7614

attached to masked objects. Therefore, we compare the prediction results with or without information leaks. In both scenarios, we ensure that the mask tokens pass through an equal number of transformer layers (i.e., one in our case) to exchange information. As shown in Tab. VII, we observe that the configuration without information leakage significantly outperforms its counterpart, affirming that the risk of information leakage outweighs the drawback of losing some positional data.

VI. CONCLUSION

To expand and diversify the scant motion data for trajectory prediction, we propose a pipeline of synthesizing driving scenes and utilizing them via pre-training to provide general representations for initializing trajectory prediction models. In this study, we efficiently generate trajectories using the original map and its augmented variant. We also evaluate strategies for optimizing data utilization, finding that self-supervised pre-training performs better than the commonly-adopted alternatives. With our synthetic data generation and self-supervised pre-training pipeline, the fine-tuned prediction model outperforms the trajectory forecasting baseline by a large margin. Our research offers a comprehensive pipeline for data generation and utilization, providing a promising direction to alleviate the data scarcity for trajectory forecasting.

VII. ACKNOWLEDGEMENT

Berkeley DeepDrive supports this work.

REFERENCES

[1] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, and D. Anguelov, "Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 9710–9719.

[2] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2636–2645.

[3] C. Xu, T. Li, C. Tang, L. Sun, K. Keutzer, M. Tomizuka, A. Fathi, and W. Zhan, "Pretram: Self-supervised pre-training via connecting trajectory and map," in *Computer Vision – ECCV 2022*, S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds. Cham: Springer Nature Switzerland, 2022, pp. 34–50.

[4] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, *et al.*, "Argoverse: 3d tracking and forecasting with rich maps," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8748–8757.

[5] M. Bahari, S. Saadatnejad, A. Rahimi, M. Shaverdikondori, A.-H. Shahidzadeh, S.-M. Moosavi-Dezfooli, and A. Alahi, "Vehicle trajectory prediction works, but not everywhere," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[6] Z.-H. Yin, C. Li, L. Sun, M. Tomizuka, and W. Zhan, "Iterative imitation policy improvement for interactive autonomous driving," *ArXiv*, vol. abs/2109.01288, 2021.

[7] Z. Wang, C. Wang, Z. Dong, and K. Ross, "Pre-training with synthetic data helps offline reinforcement learning," 2024.

[8] Z. He, G. Blackwood, R. Panda, J. McAuley, and R. Feris, "Synthetic pre-training tasks for neural machine translation," 2023.

[9] R. He, S. Sun, X. Yu, C. Xue, W. Zhang, P. Torr, S. Bai, and X. Qi, "Is synthetic data from generative models ready for image recognition?" 2023.

[10] T. Tian, C. Xu, M. Tomizuka, J. Malik, and A. Bajcsy, "What matters to you? towards visual representation alignment for robot learning," in *The Twelfth International Conference on Learning Representations*, 2024.

[11] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 07, 2020, pp. 13 001–13 008.

[12] K. Ryu, S. Hwang, and J. Park, "Instant domain augmentation for lidar semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 9350–9360.

[13] S. Kim, S. Lee, D. Hwang, J. Lee, S. J. Hwang, and H. J. Kim, "Point cloud augmentation with weighted local transformations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 548–557.

[14] K. He, R. Girshick, and P. Dollar, "Rethinking imagenet pre-training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[15] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, "Exploring the limits of weakly supervised pretraining," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[16] Y. Wei, Y. Zhang, J. Huang, and Q. Yang, "Transfer learning via learning to transfer," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research,

J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 5085–5094.

[17] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, vol. 11, no. 12, 2010.

[18] M. Assran, Q. Duval, I. Misra, P. Bojanowski, P. Vincent, M. Rabbat, Y. LeCun, and N. Ballas, "Self-supervised learning from images with a joint-embedding predictive architecture," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 15 619–15 629.

[19] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[20] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738.

[21] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 16 000–16 009.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of naacL-HLT*, vol. 1, 2019, p. 2.

[23] S. Huang, Y. Xie, S.-C. Zhu, and Y. Zhu, "Spatio-temporal self-supervised representation learning for 3d point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 6535–6545.

[24] Y. Yao, Q. Chen, A. Zhang, W. Ji, Z. Liu, T.-S. Chua, and M. Sun, "PEVL: Position-enhanced pre-training and prompt tuning for vision-language models," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 11 104–11 117.

[25] T. Chen, S. Saxena, L. Li, D. J. Fleet, and G. Hinton, "Pix2seq: A language modeling framework for object detection," in *International Conference on Learning Representations 2022*, 2022.

[26] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 8748–8763.

[27] Y. Zhang, K. Gong, K. Zhang, H. Li, Y. Qiao, W. Ouyang, and X. Yue, "Meta-transformer: A unified framework for multimodal learning," *arXiv preprint arXiv:2307.10802*, 2023.

[28] J. Wang, L. Perez, *et al.*, "The effectiveness of data augmentation in image classification using deep learning," *Convolutional Neural Networks Vis. Recognit*, vol. 11, no. 2017, pp. 1–8, 2017.

[29] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *Robotics: Science and Systems (RSS)*, 2018.

[30] H. Yisheng, W. Yao, F. Haoqiang, C. Qifeng, and S. Jian, "Fs6d: Few-shot 6d pose estimation of novel objects," *CVPR*, 2022.

[31] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, T.-Y. Lin, E. D. Cubuk, Q. V. Le, and B. Zoph, "Simple copy-paste is a strong data augmentation method for instance segmentation," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 2917–2927.

[32] W. Feng, S. Z. Zhao, C. Pan, A. Chang, Y. Chen, Z. Wang, and A. Y. Yang, "Digital twin tracking dataset (dttd): A new rgb+depth 3d dataset for longer-range object tracking applications," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2023, pp. 3288–3297.

[33] C. Azevedo, T. Gilles, S. Sabatini, and D. Tsishkou, "Exploiting map information for self-supervised learning in motion forecasting," *arXiv preprint arXiv:2210.04672*, 2022.

[34] P. Bhattacharyya, C. Huang, and K. Czarnecki, "Ssl-lanes: Self-supervised learning for motion forecasting in autonomous driving," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1793–1805.

[35] H. Chen, J. Wang, K. Shao, F. Liu, J. Hao, C. Guan, G. Chen, and P.-A. Heng, "Traj-mae: Masked autoencoders for trajectory prediction," *arXiv preprint arXiv:2303.06697*, 2023.

[36] J. Cheng, X. Mei, and M. Liu, "Forecast-mae: Self-supervised pre-training for motion forecasting with masked autoencoders," *arXiv preprint arXiv:2308.09882*, 2023.

[37] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding hd maps and agent dynamics from vectorized representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 525–11 533.

[38] P. Gao, T. Ma, H. Li, Z. Lin, J. Dai, and Y. Qiao, "Mcmae: Masked convolution meets masked autoencoders," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 35 632–35 644.

[39] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 303–15 312.

[40] M. Hallgarten, I. Kisa, M. Stoll, and A. Zell, "Stay on track: A frenet wrapper to overcome off-road trajectories in vehicle motion prediction," in *2024 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2024, pp. 795–802.