Predicting Future Behavior of Road Users

Tobias Biegert
Karlsruhe Institute of Technology
Karlsruhe, Germany
tobias.biegert@student.kit.edu

***Abstract*—In recent years, there has been a surge of interest in developing accurate and efficient trajectory prediction models for autonomous vehicles and other intelligent systems. This paper presents a comprehensive survey of state-of-the-art trajectory prediction methods, benchmarked by their minADE performance on three key datasets: nuScenes, Interaction, and Waymo Open Motion. Models are categorized based on their underlying architecture, with Graph Neural Networks and Transformers emerging as particularly effective approaches. However, the collection of top-performing models also includes methods such as reinforcement learning and novel techniques. Model performance and inference times are assessed, and current research gaps are identified. Potential new research directions are proposed, including the integration of additional information sources and the evaluation of model robustness and reliability.**

I. Introduction

The advent of autonomous driving technology holds the promise of dramatically transforming urban landscapes and preventing numerous human fatalities [36].

Forecasting the future motion of nearby agents is a crucial element in the self-driving process. It has been receiving increasing attention in recent years as it is essential for robotic vehicles to understand their surroundings and make safe decisions. Motion forecasting requires to predict future behaviors of traffic participants by considering the observed agent states as well as the scene context, which is challenging due to the inherently multimodal behaviors of the agent and complex environments.

In the past, the motion of dynamic objects has been forecasted utilizing various non-deep learning-based models such as the Kalman filter [8], linear trajectory avoidance model [23], and social force model [21].

Early deep learning-based models such as SocialLSTM [1], Precog [27] and R2p2 [26] employ Recurrent Neural Network (RNN) architectures like Long Short-Term Memory (LSTM) [14] or Gated Recurrent Unit (GRU) [4] to forecast future positions. An alternative approach is to use a Convolutional Neural Network (CNN) to predict trajectories based on a Bird's Eye View (BEV) image of the road scene [7], [15], [24].

Recently, a more sophisticated approach has been developed that employs Graph Neural Networks (GNNs) to model the interactions between agents. VectorNet [10] and LaneGCN [20] are two of the earliest models that have utilized this approach to predict motion. Additionally, Transformer-based models [34] have been very successful in Natural Language Processing, and recent trajectory prediction models have also employed this architecture.

Based on this survey of the latest state-of-the-art models, it has been observed that the majority of them have adopted either a GNN or a Transformer-based architecture.

The rest of this work is organised to a number of sections: Section II describes the methodology for identifying the relevant literature as well as the datasets used to find and compare the different approaches. It also defines the problem of trajectory prediction formally and introduces common evaluation metrics. Section III reviews the related deep learning-based solutions and classifies them. The performance of the different models is compared in section IV. Section V highlights current research gaps as well as potential new research directions and gives key concluding remarks.

II. Methodology

To identify relevant benchmarks for state-of-the-art models, it is crucial to first formulate the problem that needs to be solved.

An autonomous vehicle is assumed to be equipped with detection and tracking modules that observe state S accurately for all the involved agents A. Given a scene, the prediction target is denoted as a_{tar} and the surrounding agents are denoted as $A' = \{a_1, a_2, \ldots, a_m\}$. The state of agent $a_i \in A$ at frame t is denoted as s_i^t, including features such as position, velocity, actor type, heading angle, and $s_i = \{s_i^{-H+1}, s_i^{-H+2}, \ldots, s_i^0\}$ denotes the sequence of states sampled at different timestamps throughout the observation period H.

The aim of motion estimation is to predict the future trajectories $\tau_{tar} = \{\tau_i | i = 1, \ldots, K\}$ of the target agent a_{tar}, where $\tau_i = \{(x_i^1, y_i^1), \ldots, (x_i^T, y_i^T)\}$ denotes one of typically $K > 1$ predicted trajectories for the target agent up to the prediction horizon T.

Additionally, scene context is available in the form of a High-Definition Map (HD map) containing elements such as road shape, road markings, traffic signs, and barriers shown in Fig. 1.

A typical evaluation of a multi-agent predictive model involves generating up to K predicted trajectories for each agent. This approach is crucial since, in any given scenario, an agent may have various potential objectives, and there may be multiple paths to reach each goal. By generating multiple trajectories, the model can account for these uncertainties and provide a range of likely outcomes.

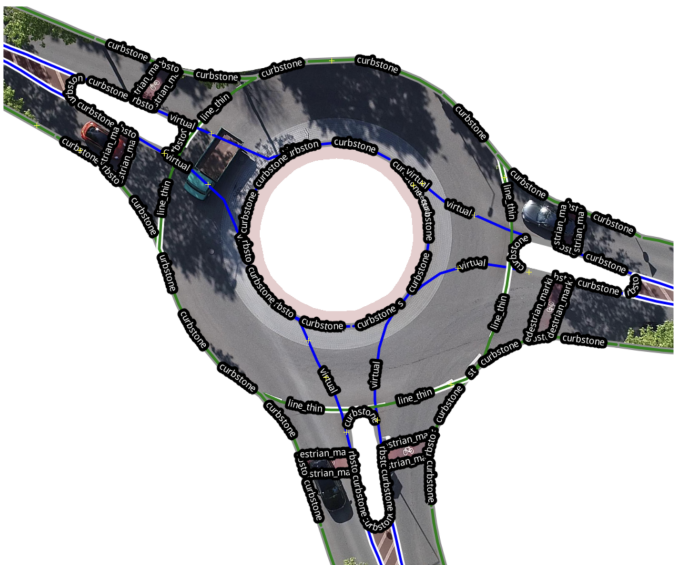

Fig. 1. Exemplary lanelet2-formatted HD map depicting a roundabout from the INTERACTION dataset [38] overlaid on a BEV image of the corresponding road.

Two common error metrics are Minimum Average Displacement Error (minADE) and Minimum Final Displacement Error (minFDE) defined as

$$minADE_K^i = \min_{k \in K} \sqrt{\frac{1}{T} \sum_{t=1}^{T} \left(x_t^{k,i} - x_t^i\right)^2 + \left(y_t^{k,i} - y_t^i\right)^2} \tag{1}$$

$$minFDE_K^i = \min_{k \in K} \sqrt{\left(x_T^{k,i} - x_T^i\right)^2 + \left(y_T^{k,i} - y_T^i\right)^2} \tag{2}$$

where (x_t^i, y_t^i) is the ground truth position of agent i and time t and $(x_t^{k,i}, y_t^{k,i})$ the predicted position according to trajectory k. Both metrics are averaged across all agents and test examples resulting in $minADE_K$ and $minFDE_K$.

Another frequently used metric is the Miss Rate which describes the ratio of the predictions where none of the forecasted trajectories are within a certain threshold, usually $2\,m$, of ground truth according to endpoint error.

A systematic approach to identify relevant literature was used. Initially, the search was narrowed down by focusing on three popular benchmark datasets: nuScenes [2], Interaction [38], and Waymo Open Motion [9]. The leaderboards for each dataset were then examined to determine the top-performing models for trajectory prediction. Specifically, models that ranked highest based on minimum Average Displacement Error (minADE) and had been published in a paper were selected. This method ensures that the models included in the review are well-performing, based on credible research and provide valuable insights into the topic of trajectory prediction for traffic participants.

The nuScenes dataset [2] includes a diverse range of sensor data from 6 cameras, 5 radars, and 1 lidar, all with a full 360-degree field of view, capturing a variety of urban driving scenarios in Boston and Singapore. The dataset provides annotations for different objects in the scene, such as cars, pedestrians, bicycles, and traffic signs, and offers a total 1000 scenes, each capturing 20 seconds of data and annotated at $2\,Hz$. Additionally, the dataset provides human-annotated semantic maps of the relevant areas. For the trajectory prediction benchmark $8\,s$ segments are devided into an observation period of $H = 2\,s$ with which the trajectories of the next $T = 6\,s$ have to be predicted.

TABLE I
COMPARISON OF THE THREE DATASETS CONSIDERED

	nuScenes	Interaction	Waymo
Total time	$5.5\,h$	$16.5\,h$	$574\,h$
Observation Period H	$2\,s$	$1\,s$	$1\,s$
Time horizon T	$6\,s$	$3\,s$	$8\,s$
Sampling rate	$2\,Hz$	$10\,Hz$	$10\,Hz$
Number of cities	2	6	6
Number of countries	2	3	1
Scene	urban	urban, highway	urban
Bounding Boxes	3D	2D	3D
Number of object types*	1	1	3
Offline labeling	human	automated	automated

*Number of object types measures the number of types of objects to predict the motion trajectory.

The INTERACTION dataset [38] captures diverse scenarios, such as signalized and unsignalized intersections, roundabouts, merging, and lane changes, collected from drone footage in North America, Europe, and Asia. The dataset contains a total of 991 minutes of motion data, captured at a resolution of $10\,Hz$, representing each trajectory as a sequence of points in time and space, along with additional information such as the velocity and heading of the vehicle at each point. The dataset also includes a corresponding semantic map in lanelet2 format, providing additional context about the scene and the lane in which the vehicle is traveling. The drone footage is post-processed offline with detection, tracking, and track smoothing. The trajectory prediction benchmark utilizes 4-second segments, dividing them into an observation period of $H = 1\,s$ to predict trajectories for the subsequent $T = 3\,s$.

The Waymo Open Motion dataset [9] contains over 100,000 scenes, each 20 seconds long at $10\,Hz$, totaling more than 570 hours of driving data. It was collected by mining for interesting interactions between vehicles, pedestrians, and cyclists across six cities within the United States. The dataset also provides corresponding high definition 3D maps for each scene as a set of polylines and polygons created from curves sampled at a resolution of 0.5 meters and 3D bounding boxes for each road agent generated by an offboard algorithm. In contrast to the other two datasets, the Waymo dataset requires predicting not only the trajectories of vehicles but also those of pedestrians and cyclists. For the trajectory prediction benchmark $1\,s$ of observations are used to predict up to $8\,s$ of future positions.

It is worth noting that all three datasets offer labeled maps of the scene, which can be a significant advantage over raw

sensor data. However, it is essential to consider that generating these labels onboard in a real-world scenario can significantly slow down inference time and potentially worsen the quality of labels due to computing power restrictions.

III. State of the Art

In this section, we will provide an overview of the various models identified using the methodology outlined in section II. A total of 15 state-of-the-art models have been selected for further investigation.

This analysis will focus on several key attributes, including data representation, encoding of agent states, roads, and interactions, as well as the decoding module responsible for generating trajectories. We will also examine other relevant attributes that may impact model performance.

A. Data Representation

1) Road Context: One of the most common methods for encoding road context is to use a BEV image of the HD map [5], [19], [28], [29], [37], which provides a straightforward representation. However, this method has several drawbacks: There is a compromise between spatial grid resolution, field of view, and computational demands. Rasterization involves manual feature engineering, and some characteristics are intrinsically challenging to represent in this structure (e.g., radial velocity). Capturing long-range interactions using convolutions with small receptive fields is challenging. The data content is highly sparse spatially, rendering a dense representation an inefficient option in terms of computation [33].

To address these issues, Gao et al. [10] propose using polylines to represent the scene context, which can be converted into sets of vectors illistrated in Fig. 2. This method has gained popularity in recent years, with 10 out of 15 state-of-the-art models representing road features this way. Typically, when using this approach, the polylines are represented as nodes of a graph where edges model the connection to nearby polylines so that the input can be used by GNNs [6], [16], [22], [39]. In particular Gilles et al. with their models GOHOME [13] and THOMAS [12], both based on their earlier model HOME [11], have adopted the polyline-based graph representation instead of the rasterized representation used in the original model. The same switch can be observed from Multipath [3] to the better performing MPA [18]. This trend indicates a shift towards using polyline representations.

Other models not using GNN layers use the polylines directly as input [30], [32].

2) Agent State: A clear trend in agent state representation can be observed among the state-of-the-art models. The majority, 14 out of 15, use sets of vectors to encode agent state features. These vectors may include, among other values, x and y position, velocity in x and y direction, timestamp, heading angle, spacial dimensions of the agent as well as his type [38]. Additionally, since the dynamics of moving agents can be approximated by polylines based on their motion trajectories [10], these polylines are typically converted into sets of vectors for ease of use. Three of these models represent the agent

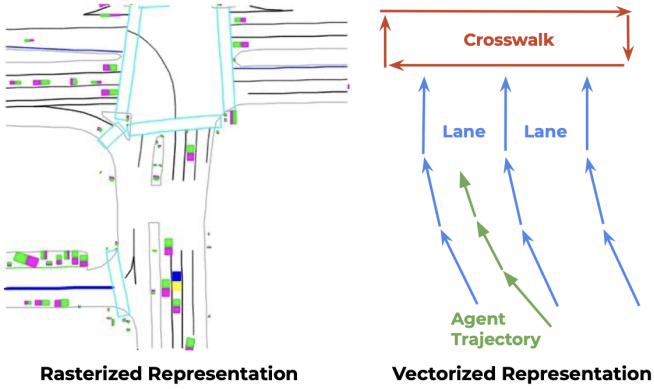

Fig. 2. Illustration of the rasterized rendering (left) and polyline approach (right) to represent HD map and agent trajectories [10].

polylines as graphs. Salzmann et al. [28] use a directed graph to model only the agent states, while Jia et al. [16] model both road features and agents in the same heterogeneous graph. Zhao et al. [39] represent polylines as subgraphs first and then integrate those in a global graph representing the entire scene.

The use of rasterized BEV images to represent agent trajectories has become less popular among state-of-the-art models. However, ITRA [29] is a recent example of a state-of-the-art model that utilizes this approach. In ITRA, the surrounding road and agents are presented to each agent as a rasterized, ego-centered, and ego-rotated BEV image.

B. Encoding

Researchers propose novel models with various architectures to improve prediction performance. This section provides a summary of common design choices regarding feature and interaction encoding of these state-of-the-art models.

Among the identified models, the two largest categories are those based on GNN architectures and those that use Transformer-based approaches.

1) Graph Neural Networks: Zhao et al. with their TNT-model [39] use the VectorNet architecture [10] to encode the entire scene. Specifically, a subgraph network is applied to encode each polyline, then a global graph is used to model the interactions between polylines. The local graphs are implemented with Multilayer Perceptrons (MLPs), and the global graphs with self-attention [34]. The output of this step is a global context feature for each modeled agent.

PGP [6] first uses three independent GRU layers to encode the target vehicle trajectory, surrounding vehicle trajectories and node features of the previously constructed graph, where nodes are lane centerlines which are connected to the next node along the lane as well as nodes in neighboring lanes. In order to incorporate nearby agent encodings, node encodings are updated using scaled dot product attention. To further refine the local context of the graph, the method aggregates information from neighboring nodes using either a GCN [17] or a GAT [35] layer. The final node encodings learned by the graph encoder are obtained from the outputs of the GNN layers. The Graph Encoder of PGP is displayed in Fig. 3.

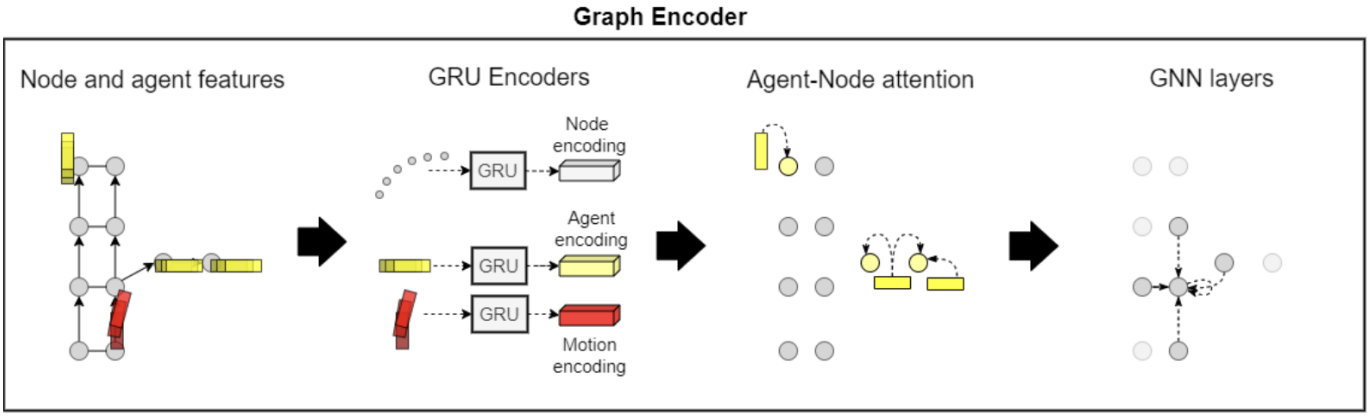

Fig. 3. The PGP Graph Encoder [6] encodes agent and map context as node encodings of a directed graph.

Both state-of-the-art models GOHOME [13] and THOMAS [12] use the same encoding structure. To encode agent features and lanelets, both models employ two separate encoders that consist of a 1D-convolution and a GRU layer. The encoded polyline features are then updated through a first GNN consisting of a sequence of four graph convolution operations. Furthermore, each encoded agent feature is updated with map information through a cross-attention layer on the lanelet features. Interactions between agents are taken into account through a self-attention layer between agents. Finally, the target agent feature is concatenated to all the lanelet features and processed through a second GNN also comprising four graph convolutions to obtain the final graph encoding used to generate predictions.

In their model Trajectron++ [28], Salzmann et al. utilize an LSTM to encode the observed agent history. To account for neighboring agents' influence on the modeled agent, edge information is first aggregated from neighboring agents of the same semantic class, with an element-wise sum serving as the aggregation operation. The aggregated states are then passed through an LSTM whose weights are shared across all edge instances of the same type. Subsequently, the encodings from all edge types that connect to the modeled node are aggregated to obtain a single "influence" representation with an additive attention module. In addition, Trajectron++ encodes a local map for each modeled agent. This map is rotated to match the agent's heading and encoded using a CNN consisting of four convolutional layers and one fully connected layer. The resulting output is concatenated with the encoded node history and edge influence representation vectors for the corresponding agent.

2) Transformers: AgentFormer [37] uses an encoder module that is similar to the original Transformer [34], but with some important changes. Instead of positional encoding, AgentFormer uses time encoding, which helps to preserve the temporal information in the input sequence. Moreover, the model introduces a new attention mechanism that is agent-aware. The model learns to attend to elements of the same agent differently than elements of other agents, which helps to preserve the notion of agent identity while still ensuring permutation invariance of agents. AgentFormer also encodes connectivity information between agents by masking out the attention weights between unconnected agents. Finally, a CNN is used to encode road context from an image that is cropped and aligned around each agent.

The Wayformer encoding module [22] consists of one or more standard Transformer encoders, with the notable difference being the use of learned positional encoding instead of standard positional encoding. The paper presents three different types of fusion techniques for combining the different input modalities: late fusion, early fusion, and hierarchical fusion. In late fusion, each modality has its own dedicated encoder. Early fusion uses a single "Cross-Modal-Encoder" to combine all modalities, while hierarchical fusion combines both approaches. The paper also explores two different types of attention: multi-axis attention, which applies self-attention across both spatial and temporal dimensions simultaneously, and factorized attention, which splits attention across the two dimensions. Additionally, the paper explores the use of latent queries, which are added to the attention mechanism to improve performance.

Both MTR and its ensemble variation MTR-A [30] use the same encoding module to model the interaction between agents and the environment in autonomous driving tasks. The encoding module first encodes the polyline representations of agents' histories and the road map using PointNet-like [25] polyline encoders. Then, a standard Transformer encoder module takes these encoded features as input and applies self-attention modules to model the interaction between agents and encode scene environment features.

The HDGT model proposed by Jia et al. [16] combines a GNN and a Transformer architecture. The model first processes ego-centric agent features and road polyline features using a 1D-CNN and a simplified PointNet with MLP, respectively. Edge features are encoded using a ViewShift unit, which shifts the view from agent u's ego-centric pose to agent v's using an MLP layer with different parameters for different source node types. The encoded features are then fed

to a GNN. To incorporate the Transformer architecture into the GNN, the authors adopt it as the aggregation function. Specifically, each node's feature is used as the query, and its in-edges' features are used as the keys and values to update the node feature. Edge features are concatenated with the source node feature, and an MLP layer is used to obtain the updated edge feature.

3) Others: Tang et al. [32], in their Golfer architecture, propose a novel core component called the Mix and Match (MnM) Block. This block generalizes the transformer by formulating the attention mechanism as a special instance of the MnM operation, which replaces the expensive attention computation with a more efficient operation. To encode agent and road features, Golfer employs stacked and multi-headed MnM blocks using MaxPool as the Mix operation and Concatenation as the Match operation. Interaction is modeled by another multi-headed MnM block. In this block, the matching operation is an element-wise multiplication between the ego feature vector and latent features of road and agent polylines. The mixing operation is MaxPool. Ego features and their interactions are then concatenated and passed through an MLP to obtain the final output of the encoding module.

With MPA [18], Konev and Stepan introduce a new type of context awareness referred to as multi-context gating (MCG), which is considered an efficient version of cross-attention. To encode agent information, the authors first pass past positions and position differences through an LSTM and then through an MCG block. Additionally, the past observations of neighboring agents are also fed into an LSTM and then fused using another MCG block. Finally, for each agent, the closest road polylines are converted into their respective frame of reference and processed by a shared MLP. The output is then fused with agent history embeddings using stacked MCG blocks. All these embeddings are then concatenated.

In the DESIRE model [19], the past trajectories are encoded using a GRU, while the scene context is encoded with a CNN. The CNN feature maps are then downsampled using a pooling operation to reduce their spatial dimensionality and capture local spatial patterns and features.

The initial component of P2T [5] consists of a reward model, composed of convolutional and pooling layers. For each cell on a large-scale 2-D grid, the reward model uses local scene context and motion features to create a transient path state reward and a terminal goal state reward based on the agent's trajectory history which can then be used by an approximate value iteration algorithm to obtain a MaxEnt policy. Moreover, the agents' track history is encoded using a GRU encoder.

ITRA [29] is built on a fully differentiable simulator and employs Conditional Recurrent Variational Neural Networks (CVRNNs) to model agents. The CVRNNs utilize GRUs in conjunction with a CNN encoder for processing BEV images. These images are embedded within the simulation, capturing the positions of agents and the environmental context.

C. Prediction

Some researchers employ regression processes and decode features with MLPs. Inspired by anchor proposals in computer vision, other researchers add a classification branch to predict probability or confidence scores for each proposed trajectory anchor. Some models use transformer decoders, while others use generative models like Conditional Variational Autoencoders (CVAEs) [31] or sampling from a policy learned using reinforcement learning or a deterministic sampling optimization to treat trajectory prediction as a conditional sampling and selection process.

1) Regression with MLP: In HDGT [16], two distinct MLPs are utilized, with one dedicated to generating predicted trajectories and the other focused on estimating the corresponding occurrence probabilities.

The Golfer model [32] utilizes a similar simple yet effective decoder design, which involves passing the encoded feature obtained from the encoding module through separate MLPs to predict a set of potential future trajectories, each accompanied by a corresponding occurrence probability. But with Golfer, the final output is produced by an ensemble of multiple models, and all outputs are collected to perform a weighted k-Means clustering. The centroid of each cluster, along with its corresponding aggregated and normalized weight, is then used to determine the predicted trajectory and its associated probability.

2) Use of Anchors: In the TNT approach [39], predefined points on lane centerlines are utilized as anchors. These and the global context feature for each agent gained from the encoding step are then used for trajectory prediction through a three-step process: First, an MLP generates a discrete probability distribution over the anchors. Next, the most probable trajectory for each anchor is generated non-autoregressively via another MLP. Finally, an additional MLP scores the resulting trajectory hypotheses and K trajectories are greedily selected by an algorithm similar to non-maximum suppression.

Shi et al. [30] propose an innovative transformer-based decoder network for predicting multimodal future trajectories in their MTR and MTR-A models. To model motion prediction, they introduce motion query pairs, each associated with a specific intention point generated using the k-means algorithm on the endpoints of ground-truth trajectories in the training set. The transformer decoder employs cross-attention to aggregate context information from both agent and map features for each motion query pair. In each layer, the queried feature for each motion query pair is used to predict the future trajectory through a prediction head with multiple MLP layers. The model uses a Gaussian Mixture Model (GMM) to represent the multimodal future motion, where a probability p and GMM parameter $N(\mu_x, \sigma_x; \mu_y, \sigma_y; p)$ for each motion query pair at each future time step are predicted. For MTR-A, a model ensemble strategy is used to combine the results from multiple variants of the framework.

The MPA model [18] is designed to learn anchor embeddings as an part of its overall training process. In addition to the concatenated encodings of agent states, road

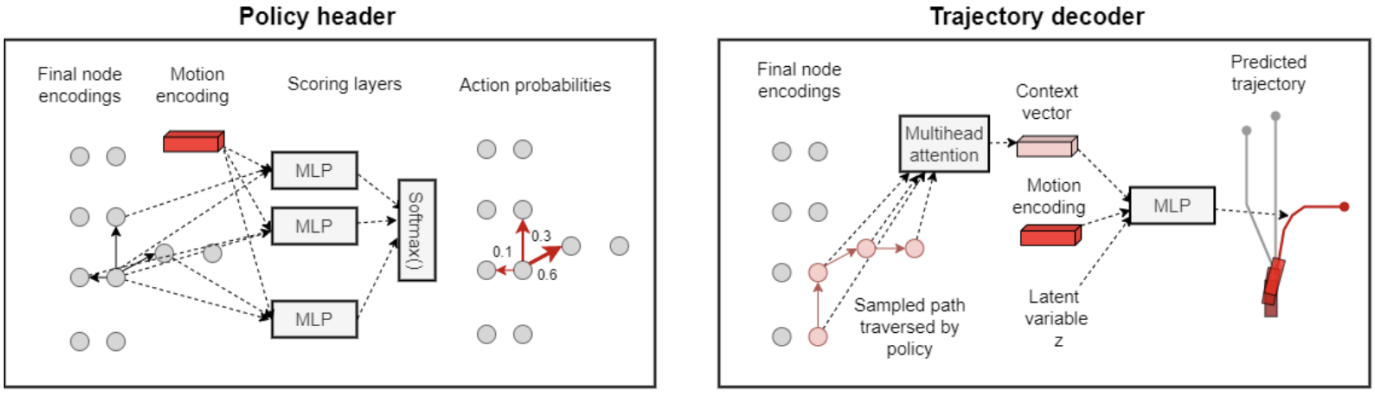

Fig. 4. The PGP [6] policy header learns a discrete policy for sampled graph traversals. The trajectory decoder predicts trajectories by selectively attending to node encodings along paths traversed by the policy and a sampled latent variable.

information, and interactions, these embeddings are fed into multiple MCG blocks. The outputs are combined using multi-head attention and MaxPooling and fed into another MCG block. Subsequently, a final MLP is employed to generate the output parameters of a GMM. To further enhance the model's predictive power, multiple predictor heads are assembled and combined through bootstrap aggregation.

3) Transformer Decoder: The Wayformer model [22] utilizes a stack of one or more standard transformer cross-attention blocks for its decoder, which takes learned initial queries and cross-attends them with the scene encoding to generate trajectories. The predictor component of Wayformer generates a mixture of Gaussians to represent the potential trajectories an agent may follow. In order to reduce the number of modes being considered while preserving the diversity of the original output mixture, the model applies trajectory aggregation following the approach employed within the MPA model [18].

4) Deterministic Sampling Optimization: Gilles et al. present their model GOHOME [13], which employs the final graph encoding produced by the encoding module to score lanes by applying a linear layer to the graph encoding, followed by a sigmoid activation function. Subsequently, a local curvilinear raster is generated for the top K lanes, which is then integrated into a predicted probability distribution heatmap. To extract the final trajectory points from the heatmap, a MissRate optimization sampling algorithm is employed. This algorithm iteratively selects the grid point with the highest surrounding probability within a local neighborhood and subsequently sets the probabilities within this neighborhood to zero. Full trajectories are then generated from the sampled endpoints using a two-layer MLP.

The prediction module of THOMAS [12] utilizes the same sampling algorithm and full trajectory generation method as GOHOME. However, it incorporates a novel, efficient hierarchical heatmap process that enables scaling for simultaneous multi-agent prediction. Additionally, THOMAS features a unique scene-consistency module that reassembles the marginal outputs into a cohesive joint prediction.

5) Sampling from Policy: PGP [6] leverages node encodings from the graph encoder and the encoded past trajectory to learn a graph traversal policy, producing likely future routes. The policy, a discrete probability distribution over outgoing edges at each node, is learned through an MLP with shared weights. Using multi-head scaled dot product attention, the trajectory decoder aggregates map and agent context over a sampled route, generating a context vector. To sample a trajectory, a policy roll-out and context vector are obtained, concatenated with a sampled latent vector, and passed through an MLP. Similar to [32], K trajectories are chosen from numerous generated ones using k-Means clustering. A visual representation of the PGP prediction mechanism is shown in Fig. 4 as an example.

Deo et al. [5] sample from their P2T model's MaxEnt policy to generate a set of diverse paths towards multiple potential goals. These sampled plans are then fed to the trajectory generator, which encodes them using a Bidirectional GRU (BiGRU). In the final step, a GRU decoder is utilized, which incorporates soft-attention and takes into account the encoded agent state history along with the encoded plans to output a set of trajectories for the prediction horizon. Again, the predicted trajectories are clustered using k-Means to obtain a set of K final trajectory predictions.

6) Conditional Variational Models: Trajectron++ [28] effectively manages multimodality by utilizing the CVAE latent variable framework. This approach generates the target distribution, which is parameterized using a bidirectional LSTM during the training phase. During inference, the representation vector obtained from the encoding process and the sampled latent variable are fed into a GRU. Each GRU cell produces the parameters for a bivariate Gaussian distribution over control actions, such as acceleration and steering rate. The agent's system dynamics are subsequently integrated with these control actions, resulting in trajectories.

In the DESIRE framework [19], the integration of a CVAE and an GRU decoder facilitates the generation of diverse and feasible future trajectories. During the inference phase, the CVAE samples multiple latent variables from the learned

distribution, with each sampled latent variable representing a distinct possible mode for future trajectories. The GRU decoder utilizes the representation vector obtained from the encoder, along with the sampled latent variables, as its input. Subsequently, it produces a series of future trajectory points for every latent variable. These generated trajectories are then ranked and selected based on their likelihood within an inverse optimal control framework, which also takes into account the encoded semantic scene context.

AgentFormer [37] employs a CVAE-based trajectory sampler and a future trajectory decoder. The trajectory sampler generates K sets of latent codes, with each set containing the latent codes for all agents, which can be decoded by into a multi-agent future trajectory sample. Initially, a feature sequence is created by concatenating the currently generated trajectories with their corresponding latent codes. This feature sequence is then combined with the encoded semantic map features and timestamped before being used as queries for the AgentFormer decoder, along with the encoded past feature sequence that serves as keys and values. Each element of the output sequence is subsequently passed through a MLP to produce the decoded future agent position. Distinct from the original Transformer decoder, the AgentFormer future trajectory decoder is autoregressive, meaning it generates trajectories one step at a time and feeds the current trajectories back into the model to produce the trajectories for the next time step. The last position feature serves as the initial sequence in this autoregressive process.

ITRA [29] utilizes CVRNNs to generate steering and acceleration outputs for each agent based on the respective BEV image and agent state. These outputs are subsequently fed into a kinematic bicycle model, which computes the agents' future states based on the predicted actions. Finally, the complete simulation state, encompassing the updated agent states, undergoes differentiable rendering. This process enables gradient-based optimization during training and streamlines the generation of the next time step, enhancing the model's ability to predict multi-agent trajectories effectively.

IV. Evaluation

A. Performance

Table II presents the evaluation of various models on the nuScenes dataset, ranked by their minADE. Alongside minADE, the table also reports minFDE and Miss Rate. For evaluating minADE and Miss Rate, five trajectories were used, while a single trajectory was employed for calculating minFDE. $2\,m$ is used as the threshhold for Miss Rate. The top three positions are dominated by GNNs, with PGP [6] emerging as the best-performing model. Gilles et al. successfully enhance the performance of the GOHOME model [13] by introducing their novel THOMAS model [12], which surpasses GOHOME in all metrics and stands as the top model concerning minFDE. Furthermore, P2T [5] and AgentFormer [37] secure the 4th and 5th ranks, representing top-performing models based on reinforcement learning and Transformer approaches, respectively.

TABLE II
Evaluation on nuScenes Dataset

	$minADE_5$	$minFDE_1$	$MR_{5,2}$
PGP [6]	1.27	7.17	0.52
THOMAS [12]	1.33	6.71	0.55
GOHOME [13]	1.42	6.99	0.57
P2T [5]	1.45	10.5	0.64
AgentFormer [37]	1.86	-	-
Trajectron++ [28]	1.88	9.52	0.70

TABLE III
Evaluation on Interaction Dataset

	$minADE_6$	$minFDE_6$
ITRA [29]	0.17	0.49
TNT [39]	0.21	0.67
DESIRE [19]	0.32	0.88
GOHOME [13]	-	0.45

Table III summarizes the performance of four models on the Interaction dataset, based on their minADE and minFDE metrics. Unlike the nuScenes dataset, the evaluation is based on six trajectories for both metrics. ITRA [29] emerges as the top-performing model, achieving the lowest minADE and minFDE scores of 0.17 and 0.49, respectively. TNT [39] and DESIRE [19] follow ITRA in the rankings, with minADE scores of 0.21 and 0.32, and minFDE scores of 0.67 and 0.88. GOHOME [13] is the only model in the table that does not report minADE. However, it is worth noting that GOHOME achieves a competitive minFDE score of 0.45, which is better than the minFDE scores of TNT [39] and DESIRE [19]. It is noteworthy that no Transformer-based models achieved a high ranking in the INTERACTION benchmark. In contrast, the model that performed the best was the only one to include a kinematic model component.

GOHOME is the only model evaluated on more than one of the benchmarks.

Table IV presents the performance evaluation of several models on the Waymo Open Motion dataset, measured by their minADE, minFDE, and Miss Rate. The evaluation is based on six trajectories for all metrics, and the Miss Rate threshold is set at $2\,m$. Wayformer [22] with both factorized and multi-axis attention outperforms other models and ranks at the top two positions in terms of both minADE and minFDE. Golfer [32] and MTR-A [30] follow closely behind in the rankings, with similar scores for both metrics. HDGT [16], MPA [18], and MTR [30] are the models with relatively lower performance in the table, with higher minADE and minFDE scores than the top four models. MTR-A exhibits the lowest Miss Rate and can outperform its non-ensemble variant MTR in every metric.

B. Inference Time

To deploy a model successfully in real-world scenarios, it must generate predictions with low latency and on limited

TABLE IV
EVALUATION ON WAYMO OPEN MOTION DATASET

	$minADE_6$	$minFDE_6$	$MR_{6,2}$
Wayformer factorized [22]	0.5447	1.1255	0.1229
Wayformer multi-axis [22]	0.5454	1.1280	0.1228
Golfer [32]	0.5533	1.1608	0.1354
MTR-A [30]	0.5640	1.1344	0.1160
HDGT [16]	0.5703	1.1434	0.1440
MPA [18]	0.5913	1.2507	0.1603
MTR [30]	0.6050	1.2207	0.1351

equipment. Out of the 15 state-of-the-art models surveyed, only five report their inference times. THOMAS, for example, can predict for 32 concurrent agents in just $20\,ms$ and for 128 agents in $31\,ms$, outperforming GOHOME, which requires $36\,ms$ and $90\,ms$ in the same scenarios. This highlights the superiority of the hierarchical decoder in improving latency, although the authors do not provide any hardware information. P2T, on the other hand, requires $79\,ms$ of inference time when run on an NVIDIA GeForce GTX 1080 Ti GPU. TNT, which utilizes an NVIDIA v100 GPU, can complete a scenario with 5 agents in just $10.8\,ms$ and one with 50 agents in $64.4\,ms$. Wayformer opts for a straightforward architecture, which allows it to achieve latencies as low as $4\,ms$ while still outperforming all other models on the Waymo Open Motion benchmark. Nayakanti et al. provide the most comprehensive evaluation of their model's latency, demonstrating its dependence on the form of attention and fusion used.

V. CONCLUSION

In conclusion, this paper has provided an overview of state-of-the-art models for multi-agent trajectory prediction, with a focus on three benchmark datasets: nuScenes, Interaction, and Waymo Open Motion. We have seen that recent advances in deep learning, particularly the use of graph neural networks and attention mechanisms, have significantly improved the performance of trajectory prediction models. However, there are still several research gaps and challenges that need to be addressed.

One significant challenge is the lack of diversity in the datasets used for evaluation. Most of the existing benchmarks focus on predicting trajectories for vehicles, pedestrians, and cyclists in urban environments, but there is a need for more diverse datasets that cover a broader range of scenarios and types of agents. Another challenge is the need to develop models that can generalize well to new environments and scenarios. Currently, most models are trained and evaluated on specific datasets, and their performance may deteriorate when applied to different environments or scenarios. Some of the trained models are openly available and could be evaluated on multiple benchmarks.

New research directions in trajectory prediction should focus on the integration of additional sources of information, such as signaling lights, since they are crucial for human interaction and navigation. Another important research direction is the development of methods to evaluate the robustness and reliability of trajectory prediction model, as their performance can have significant implications for the safety of autonomous vehicles and other intelligent systems.

In summary, while significant progress has been made in multi-agent trajectory prediction, there still remain several research gaps and challenges that require further investigation. The development of more diverse datasets, models that can generalize well to new environments, and methods for integrating additional information sources and evaluating model robustness are all promising areas for future research. By addressing these challenges, we can further improve the accuracy and reliability of trajectory prediction models and enable the development of safer and more efficient autonomous systems.

REFERENCES

[1] ALAHI, A., GOEL, K., RAMANATHAN, V., ROBICQUET, A., FEI-FEI, L., AND SAVARESE, S. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 961–971.

[2] CAESAR, H., BANKITI, V., LANG, A. H., VORA, S., LIONG, V. E., XU, Q., KRISHNAN, A., PAN, Y., BALDAN, G., AND BEIJBOM, O. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2020), pp. 11621–11631.

[3] CHAI, Y., SAPP, B., BANSAL, M., AND ANGUELOV, D. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449* (2019).

[4] CHO, K., VAN MERRIËNBOER, B., BAHDANAU, D., AND BENGIO, Y. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[5] DEO, N., AND TRIVEDI, M. M. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735* (2020).

[6] DEO, N., WOLFF, E., AND BEIJBOM, O. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning* (2022), PMLR, pp. 203–212.

[7] DJURIC, N., RADOSAVLJEVIC, V., CUI, H., NGUYEN, T., CHOU, F.-C., LIN, T.-H., SINGH, N., AND SCHNEIDER, J. Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (2020), pp. 2095–2104.

[8] ESS, A., SCHINDLER, K., LEIBE, B., AND VAN GOOL, L. Object detection and tracking for autonomous navigation in dynamic environments. *The International Journal of Robotics Research 29*, 14 (2010), 1707–1725.

[9] ETTINGER, S., CHENG, S., CAINE, B., LIU, C., ZHAO, H., PRADHAN, S., CHAI, Y., SAPP, B., QI, C. R., ZHOU, Y., ET AL. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 9710–9719.

[10] GAO, J., SUN, C., ZHAO, H., SHEN, Y., ANGUELOV, D., LI, C., AND SCHMID, C. 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. 11525–11533.

[11] GILLES, T., SABATINI, S., TSISHKOU, D., STANCIULESCU, B., AND MOUTARDE, F. Home: Heatmap output for future motion estimation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)* (2021), IEEE, pp. 500–507.

[12] GILLES, T., SABATINI, S., TSISHKOU, D., STANCIULESCU, B., AND MOUTARDE, F. Thomas: Trajectory heatmap output with learned multi-agent sampling. *arXiv preprint arXiv:2110.06607* (2021).

[13] GILLES, T., SABATINI, S., TSISHKOU, D., STANCIULESCU, B., AND MOUTARDE, F. Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 International Conference on Robotics and Automation (ICRA)* (2022), IEEE, pp. 9107–9114.

[14] HOCHREITER, S., AND SCHMIDHUBER, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[15] HONG, J., SAPP, B., AND PHILBIN, J. Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), pp. 8454–8462.

[16] JIA, X., WU, P., CHEN, L., LI, H., LIU, Y., AND YAN, J. Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. *arXiv preprint arXiv:2205.09753* (2022).

[17] KIPF, T. N., AND WELLING, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[18] KONEV, S. Mpa: Multipath++ based architecture for motion prediction. *arXiv preprint arXiv:2206.10041* (2022).

[19] LEE, N., CHOI, W., VERNAZA, P., CHOY, C. B., TORR, P. H., AND CHANDRAKER, M. Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 336–345.

[20] LIANG, M., YANG, B., HU, R., CHEN, Y., LIAO, R., FENG, S., AND URTASUN, R. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16* (2020), Springer, pp. 541–556.

[21] LUBER, M., STORK, J. A., TIPALDI, G. D., AND ARRAS, K. O. People tracking with human motion predictions from social forces. In *2010 IEEE international conference on robotics and automation* (2010), IEEE, pp. 464–469.

[22] NAYAKANTI, N., AL-RFOU, R., ZHOU, A., GOEL, K., REFAAT, K. S., AND SAPP, B. Wayformer: Motion forecasting via simple & efficient attention networks. *arXiv preprint arXiv:2207.05844* (2022).

[23] PELLEGRINI, S., ESS, A., AND VAN GOOL, L. Predicting pedestrian trajectories. *Visual Analysis of Humans: Looking at People* (2011), 473–491.

[24] PHAN-MINH, T., GRIGORE, E. C., BOULTON, F. A., BEIJBOM, O., AND WOLFF, E. M. Covernet: Multimodal behavior prediction using trajectory sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020), pp. 14074–14083.

[25] QI, C. R., SU, H., MO, K., AND GUIBAS, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 652–660.

[26] RHINEHART, N., KITANI, K. M., AND VERNAZA, P. R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)* (2018), pp. 772–788.

[27] RHINEHART, N., MCALLISTER, R., KITANI, K., AND LEVINE, S. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 2821–2830.

[28] SALZMANN, T., IVANOVIC, B., CHAKRAVARTY, P., AND PAVONE, M. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16* (2020), Springer, pp. 683–700.

[29] ŚCIBIOR, A., LIOUTAS, V., REDA, D., BATENI, P., AND WOOD, F. Imagining the road ahead: Multi-agent trajectory prediction via differentiable simulation. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)* (2021), IEEE, pp. 720–725.

[30] SHI, S., JIANG, L., DAI, D., AND SCHIELE, B. Mtr-a: 1st place solution for 2022 waymo open dataset challenge–motion prediction. *arXiv preprint arXiv:2209.10033* (2022).

[31] SOHN, K., LEE, H., AND YAN, X. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems 28* (2015).

[32] TANG, X., ESHKEVARI, S. S., CHEN, H., WU, W., QIAN, W., AND WANG, X. Golfer: Trajectory prediction with masked goal conditioning mnm network. *arXiv preprint arXiv:2207.00738* (2022).

[33] VARADARAJAN, B., HEFNY, A., SRIVASTAVA, A., REFAAT, K. S., NAYAKANTI, N., CORNMAN, A., CHEN, K., DOUILLARD, B., LAM, C. P., ANGUELOV, D., ET AL. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)* (2022), IEEE, pp. 7814–7821.

[34] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, Ł., AND POLOSUKHIN, I. Attention is all you need. *Advances in neural information processing systems 30* (2017).

[35] VELIČKOVIĆ, P., CUCURULL, G., CASANOVA, A., ROMERO, A., LIO, P., AND BENGIO, Y. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[36] WOENSEL, L. V., AND ARCHER, G. Ten technologies which could change our lives: Potential impacts and policy implications in-depth analysis.

[37] YUAN, Y., WENG, X., OU, Y., AND KITANI, K. M. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 9813–9823.

[38] ZHAN, W., SUN, L., WANG, D., SHI, H., CLAUSSE, A., NAUMANN, M., KUMMERLE, J., KONIGSHOF, H., STILLER, C., DE LA FORTELLE, A., ET AL. Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps. *arXiv preprint arXiv:1910.03088* (2019).

[39] ZHAO, H., GAO, J., LAN, T., SUN, C., SAPP, B., VARADARAJAN, B., SHEN, Y., SHEN, Y., CHAI, Y., SCHMID, C., ET AL. Tnt: Target-driven trajectory prediction. In *Conference on Robot Learning* (2021), PMLR, pp. 895–904.