

TrajGNAS: Heterogeneous Multiagent Trajectory Prediction Based on a Graph Neural Architecture Search

Yunheng Xu, Jie Chen, Shuoheng Wang, Xinwen Wang
 Anhui University, China
 xyhshen@163.com

Abstract

Most existing deep learning-based trajectory prediction algorithms rely on manual experience to design a prediction model that matches a specific task or scenario by repeatedly and manually adjusting its structure and parameters. This approach is not only complicated and inefficient to implement but also makes it difficult to balance the achieved inference speed and prediction accuracy. To solve this problem, a heterogeneous multiagent trajectory prediction technique based on a graph neural architecture search, referred to as TrajGNAS, is innovatively proposed for the first time; this method is able to automatically perform an end-to-end graph architecture search to obtain the optimal trajectory prediction model. To improve the interpretability of the model and its ability to heterogeneously perceive different scenarios, we design a bootstrapping mechanism based on physical and risk interactions as a way to guide the architecture search process. In addition, we construct a new Neural Architecture Search loss function called SocialMI-Loss, which integrates various factors, such as accuracy, a posteriori knowledge, and model complexity, to attain a balance between accuracy and computational complexity during the learning process of the model. A series of comparative experiments conducted on two large-scale automated driving datasets (nuScenes and Argoverse) demonstrate the superiority of our method. The experimental results show that TrajGNAS performs comparably to the state-of-the-art methods, but its searched architectures still possess lightweight characteristics.

1. Introduction

When constructing safe and efficient autonomous driving systems as well as smart cities, it is crucial to accurately and quickly predict the future trajectories of surrounding road users. Multiagent trajectory prediction (MATP) [18][36] aims to predict the future trajectories of multiple agents in a driving scenario based on the current and historical loca-

tion information of the traffic participants, along with the traffic situation or the intentions and behavioral patterns of the agents. Graph neural networks (GNNs) have been adopted by most trajectory prediction algorithms because of their excellent ability to understand the interactions of agents and model the semantic context of a scene [41][22]. The early GNN-based trajectory prediction models usually focused only on the interactions between traffic participants belonging to a single type to predict their trajectories due to data volume limitations. In recent years, with the release of large-scale autonomous driving datasets such as Waymo [34], Nuscenes[1], and Argoverse [3] has led to more diverse trajectory prediction scenarios and data types. Trajectron++[33] is the seminal heterogeneous multiagent trajectory prediction algorithm, which obtains the trajectory prediction information of heterogeneous traffic participants by separately encoding different types of agents to obtain the features of the heterogeneous bodies contained in the given scene. While other methods like HRG-HSG [8] incorporate road semantics to model the impact of traffic rules and layouts, thus enabling trajectory prediction for heterogeneous agents.

However, most existing methods rely on manually designed neural network architectures, and this design approach often suffers from the following difficulties.

1. The neural network design process relies on large amounts of manual work and professional experience, and to design a satisfactory network, it is often necessary to constantly experiment with and adjust the constructed network. This process is tedious and time-consuming, and the designed network often fails to adequately explore the effects of different network architectures and hyperparameters on the accuracy of the resulting model, which is usually only locally optimal.
2. As the amounts of data information and features that are available in a dataset increase, manually designed neural networks often struggle to balance their inference speeds and accuracy levels, thus posing a challenge for edge deployments with limited computational resources.

To solve the above problems, researchers have turned

their attention to automatic machine learning (AutoML), which is an automated and data-driven approach that aims to autonomously design of neural networks by automating the data preprocessing, feature selection, model selection and hyperparameter optimization procedures. The graph neural architecture search (GNAS)[11] is an NAS method designed for graph neural networks that has significant advantages in terms of processing heterogeneous graph information. By optimizing the aggregation operator and information transfer mechanism underlying the search process, the GNAS is able to construct efficient and heterogeneous-aware graph neural networks, yielding significantly improved task performance. Based on the GNAS, a novel end-to-end framework for trajectory prediction is proposed in this paper; this method fuses search strategies involving physical and social behavioral constraints, optimizes the search direction, and thus generates a network architecture with excellent heterogeneous sensing capabilities, this is fundamentally different from the previous work.

Therefore, to satisfy the demand for heterogeneous agents during the task of multiagent trajectory prediction and to improve the adaptability and automation levels of deep trajectory prediction models in multiple scenarios, inspired by the idea of NAS, we innovatively propose a heterogeneous multiagent trajectory prediction method based on the GNAS, called TrajGNAS. This method can be used to predict the trajectories of multiple intelligent bodies based on the heterogeneous input data derived from multiple interacting agents of different semantic types as well as physical and social behavioral constraints. It automatically performs an end-to-end search to find the optimal graph network model for trajectory prediction, and this model ultimately generates multimodal predicted trajectories. The main contributions of this paper are summarized as follows:

1. A novel trajectory prediction framework based on a graph neural architecture search is proposed, with an innovative design consisting of a flexible, customizable and efficient search space, a supernet search strategy, and a subnetwork contribution evaluation strategy. TrajGNAS is able to autonomously perform end-to-end search tasks to produce advanced trajectory prediction models with strong performance. To the best of our knowledge, TrajGNAS is the first work to use graph neural architecture search to optimize trajectory prediction tasks.
2. To adapt to different autonomous driving scenarios with trajectory prediction tasks and to enhance the interpretability of the constructed model, a learnable guidance mechanism based on physical and risky interaction patterns is integrated into the search. This mechanism mainly acquires heterogeneous kinematic information, such as position, velocity and orientation angle information, and combines it with a collision risk calculation

method to steer the network architecture search on the basis of the heterogeneous physical information.

3. To consider the actual deployment efficiency and safety level of the model, we design a new NAS loss function that integrates the road semantic layout, utility constraints, model computational complexity constraints, and a multimodal prediction trajectory accuracy loss to optimize the balance between the inference speed and model accuracy.

2. Related Work

Heterogeneous Multiagent Trajectory Prediction. Graph neural networks (GNNs) have been proven to have excellent performance in terms of handling heterogeneous information networks (HINs) [21] [35], and they have significant advantages in trajectory prediction tasks, especially when modeling complex interactions among agents and dynamic environments. GNNs are able to solve the problem effectively concerning complex interactions in multiple intelligent systems by virtue of their information propagation mechanisms, flexible structured data representation capabilities, and powerful generalizability. They can address complex interactions in multi-intelligent body systems and have thus become some of the key technologies in this field. In a heterogeneous map of driving scenarios, nodes generally consist of traffic participants (pedestrians, vehicles, and bicycles) and map elements (lanes, traffic signals, speed limit signs, etc.). Owing to the constraints imposed traffic rules and collision avoidance goals, there are usually explicit or implicit interactions between these traffic entities, which form the edge connections of the graph.

Trajectron++ [33] is a typical heterogeneous graph interaction model that captures the properties of different types of agents by independently encoding them via different GNN architectures and parameters. The work[10] further investigated the complexity and variability of interactions in dynamic scenes and designed a heterogeneous graph convolutional recurrent network to capture the dynamic evolution trends of heterogeneous interactions. In addition, vector-based methods have produced significant results on several large public datasets in recent years. Unlike methods that use rasterized images to represent map elements, vectorized methods learn uniform feature embeddings by extracting vectorized representations of the dynamics of multiple intelligent bodies and scene contexts. VectorNet[9] and LaneGCN[23] are pioneering methods for learning vector representations based on graph neural networks. They represent map elements as a collection of folds defined by control points and their attributes and use graph neural networks to extract features and obtain embedding vectors for trajectory prediction purposes.

Graph Neural Architecture Search. Model hyperparameter optimization is an important way to improve arti-

cial intelligence systems through automatic machine learning techniques. Neural architecture search is a branch of AutoML that aims to design efficient neural networks for complex learning tasks. The graph neural architecture search process [30] operates on the basic principles of NAS to design specific search spaces and search strategies for GNNs and optimize their hyperparameters and structures to solve the resource limitation challenges and model generalizability challenges associated with manually designing model structures. GraphNAS [11] and Auto-GNN [45] made the first attempts to solve GNN architectures by using the NAS approach. SANE [16] and DiffMG [7] focus on the two key aspects of message extraction and message passing in graph neural networks and designed NAS spaces to automatically find the optimal architecture. AutoSTG [31] extends the graph neural architecture search process to the spatiotemporal graph domain for traffic flow prediction. Although the GNAS has made progress in several domains, no research has yet applied it to the task of multi-intelligent body trajectory prediction. This gap provides great research potential for exploring GNAS-based trajectory prediction methods.

3. Method

This section details the basic concepts of the proposed TrajGNAS method. The flowchart of TrajGNAS, which consists of the elements presented below, is shown in Fig. 1.

3.1. Problem Definition

The heterogeneous multi-intelligent body trajectory task aims to generate multiple possible trajectory distributions for a variable number N_t of agents $A_1, \dots, A_{N_t}$ in a transportation scenario within a future timeframe T_{pred} based on the information observed in the previous timeframe T_{obs} . The input consists of two parts: the historical states of all the agents $A_{1, \dots, N_t}^{(t-T_{obs}:t)}$ and the map element information S . The historical states of the agents refer to the coordinates, speeds, and heading angles of the agents in the timeframe T_{obs} . In addition, the bounding box size and semantic type of the smart body (vehicle, pedestrian, or cyclist) are given. Map elements are defined as objects and signs around the intelligent body A_i , reflecting its traffic rules; they have a nonnegligible influence on the movement of the agent. These map elements usually include lanes, sidewalks, traffic signals, crosswalks and speed bumps. Each map element contains a polyline or polygon to represent its physical location as well as other information. A map element S is denoted as $\{S_1, \dots, S_{obs}\}$. Therefore, the heterogeneous multi-intelligent body trajectory task can be represented as follows:

$$\hat{Y} = \mathcal{F}(A_{obs}, S_{obs}), \quad (1)$$

where $\hat{Y}$ is the set of future trajectories predicted for all the agents in the scene, A_{obs} is the observed historical information of the agents, and S_{obs} is the information concerning the map elements. $\mathcal{F}(\cdot)$ denotes the prediction network.

3.2. Search Space

Prior to performing trajectory prediction, it is often necessary to model a heterogeneous scene graph $G = (V, E)$, where the nodes V denote heterogeneous bodies (e.g., agents, lanes, or stop signs), and the edges E denote the interactions between them. In general, GNN models are unified by a neighborhood aggregation or message-passing model [7], where the representation of each node is learned by iteratively aggregating the embeddings ("messages") of its neighbors. In the neighborhood aggregation model, a typical K-layer GNN can be represented as shown below:

$$\mathbf{h}_v^l = \sigma \left(\mathbf{W}^l \cdot \text{Aggregator}_{node}^l \left(\left\{ \mathbf{h}_u^{l-1}, \forall u \in \tilde{N}(v) \right\} \right) \right), \quad (2)$$

$$\mathbf{z}_v = \text{Aggregator}_{layer}(\mathbf{h}_v^1, \dots, \mathbf{h}_v^K), \quad (3)$$

where $\mathbf{h}_v^l$ represents the hidden features of node v learned in layer l of the network, σ denotes the nonlinear activation function, $\mathbf{W}^l$ represents the trainable weight matrix of layer l , and $\mathbf{z}_v$ signifies node v . The predefined aggregation function is the key component, which varies across GNNs used as convolution operators. Designing $\text{Aggregator}_{node}^l$ and $\text{Aggregator}_{layer}$ is usually the focus of manually designed graph neural networks, and many GNN-based trajectory prediction models choose to stack multiple layers of the same aggregation function to aggregate neighboring features. However, it has been shown [16] that combining different aggregation functions can further improve the performance of graph networks. Therefore, as shown in Fig. 1, we choose to focus on the search for the aggregation function in the graph network.

The search space of TrajGNAS consists of a number of encapsulated convolutional operator modules, called cells, that participate in the computation performed in each search layer; ultimately, each search layer determines a convolutional operator as its search result. Unlike the previously developed approaches for searching for graph neural architectures, we focus more on designing an efficient NAS algorithm that is adapted to the trajectory prediction task, aiming to automatically search for deep learning architectures that strike a balance between accuracy and speed. Our designed search space is shown in Table 1.

3.3. Risk-Aware Guided Differentiable Architecture Search

3.3.1. Risk Interaction Perception

To account for the effect of risk in the aggregation function search of the model, we choose to model collision risk as

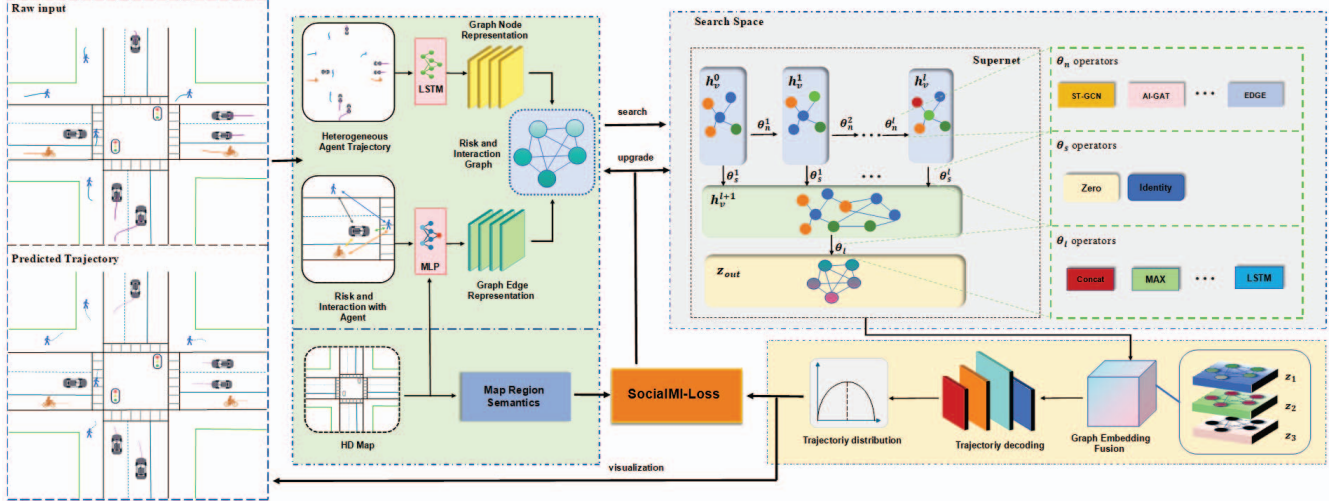


Figure 1. Overview of our proposed TrajGNAS architecture. We first use LSTM and MLP networks to encode the trajectory information and collision risk of the intelligent components, respectively, while incorporating road semantics to construct a heterogeneous graph structure consisting of risks and interactions. Next, we perform a graph neural network architecture search on this heterogeneous graph structure to select the optimal architecture. During the training process, we continuously update the architecture parameters and model weights to obtain a graph embedding representation. Finally, a multimodal trajectory distribution is generated by a trajectory decoder, from which the predicted trajectories are extracted. Moreover, we input this trajectory distribution as a posteriori knowledge into a specifically designed loss function to guide the subsequent architecture search and model training procedures.

a graph structure. Specifically, we consider the representations of the agents at moment t as nodes, and the node representations of traffic participants are usually modeled by their physical characteristics, such as their positions, speeds, facing angles, and sizes. We use a simple and efficient Long Short-Term Memory (LSTM) architecture to model this information because LSTM performs well in long time series prediction tasks and is lightweight. This process can be represented as follows:

$$h_t^a = \text{LSTM}(a_1, a_2, \dots, a_j), \quad (4)$$

where h_t^a is the final node representation at time t and $\text{LSTM}()$ denotes the LSTM network constructed by the fully connected layer and the ReLU function. $(a_1, a_2, \dots, a_j)$ are the physical attributes embedded for all the agents, i.e., their positions, velocities, facing angles, and sizes, at the current time step.

For the edge representation of the graph, we use collision risk as an edge feature. The distances, attributes, bias angles and speeds between agents are all key factors for estimating collision risk. Therefore, we quantify the collision risk between agents as shown below:

$$R_t^{ij} = \frac{1}{|T^{ij}|} * A_t^{ij} * M_t^{ij}, \quad (5)$$

where $|T^{ij}|$ denotes the collision time risk, A_t^{ij} denotes the movement direction risk, and M_t^{ij} is the agent attribute risk. The details are as follows.

Collision Time Risk: Referring to the collision time calculation method introduced in [19], the collision time is inversely related to the collision risk. Therefore, we use $\frac{1}{|T^{ij}|}$ to denote the collision risk factor with defining $|T^{ij}|$:

$$|T_t^{ij}| = \frac{\|l_t^i - l_t^j\|_2}{|v^i * \cos(|\alpha - \gamma|) - v^j * \cos(|\beta - \gamma|)|}, \quad (6)$$

where l_t^i and l_t^j denote the coordinates of agent i and agent j , respectively, at moment the t , α and β denote the angles between their forward directions and the horizontal axis, γ denotes the angle between the connecting line of the two agents and the horizontal axis, and v^i and v^j denote the speeds of agents i and j , respectively.

Facing Angle Risk: When an intelligent body i enters the field-of-view range of another intelligent body j , we determine that there is a risk of collision between them in the moving direction. The movement direction risk can be expressed as follows:

$$A_t^{ij} = \begin{cases} |\Delta_t| * \mu & \gamma \in [\alpha - \frac{\pi}{2}, \alpha + \frac{\pi}{2}] \\ 0 & \text{else} \end{cases} \quad (7)$$

$$\Delta_t = \cos(|\alpha - \gamma|) - \cos(|\beta - \gamma|), \quad (8)$$

Agent Attribute Risk: Different types of traffic participants usually have different physical attributes (speeds, masses, and volumes) and social attributes (whether they

Operator	Characteristics
GCN	• Spectral graph convolution • 1D edge feature processing
GAT	• Multi-head attention mechanism • Binary edge feature processing
ST-GCN	• Spatiotemporal joint modeling • Dynamic relationship capture
AI-GAT	• Heterogeneous weight construction • Semantic-enhanced attention
EDGE	• Multidimensional edge feature fusion • Connection enhancement mechanism
Non-GNN	• Parameter-free computation • Information passthrough
CONCAT	• Channel-wise concatenation • Message preservation mechanism
MAX	• Feature-level max-pooling • Saliency filtering
LSTM	• Gated recurrent unit • Cell state maintenance
ZERO	• Dynamic pathway pruning • Hard-gating mechanism
Identity	• Residual connection • Information highway

Table 1. Core Characteristics of Graph Operators

are vulnerable, whether they are protected by traffic rules, etc.), and these factors affect their road risk assessments. The agent attribute risk can be expressed as follows:

$$M_t^{ij} = \text{MLP} \left(h_t^i, h_t^j, m_i, m_j, c_{type} \right), \quad (9)$$

where h_t^i and h_t^j are the node representations obtained via Eq.(4), which we use here to characterize the physical attributes they contain. The m_i and m_j are the qualities of agent i and agent j , respectively, and c_{type} is a categorical variable for representing the type of agent, indicating the specific risk coefficients for different types of traffic participants.

3.3.2. Differentiable Architecture Search

For multilayer GNNs, the search space usually consists of aggregation operators and jump operations applied at multiple nodes, and our goal is to select the appropriate operator at each edge to produce a discrete search space. Inspired by the structure of DARTS [24], we relax the categorical choices regarding a particular operation to a softmax representation of all possible operations:

$$\bar{o}^{ij}(x) = \sum_{o \in \mathcal{O}} \frac{\exp(\theta_o^{ij})}{\sum_{o' \in \mathcal{O}} \exp(\theta_{o'}^{ij})} O(x), \quad (10)$$

By introducing the mixing weights computed via the softmax function, we can backpropagate the operator mixing weights θ_o^{ij} and update them in the hypernet to finally obtain the output features. This process can be expressed as follows:

$$\mathbf{z}_{out} = \theta_l \left(\theta_s \left(\mathbf{h}_v^1 \right), \dots, \bar{o}_s \left(\mathbf{h}_v^l \right) \right), \quad (11)$$

The task of the TrajGNAS architecture search is to learn continuous variables $\theta_n, \theta_l, \theta_s$ and network weights $\mathbf{w}$. The architecture parameters θ are optimized on the validation set, and the network weights $\mathbf{w}$ are optimized on the training set. Notably, when the architectural parameters and network weights are optimized, we search for the completed risk interaction perception graph constructed in Section 3.3.1 and use a predesigned loss function with task adaptation. This loss function is described in detail in the next section. We use this approach to guide the search for the optimal architecture by combining agent history trajectories, agent interaction mechanisms, and collision risks at the hypergrid feature level.

3.4. Multiple-Indicator Loss Function: SocialMI-Loss

In the real world, the behaviors of traffic participants are usually governed by traffic rules and social ethics. For example, vehicles should drive in motorized lanes, pedestrians should walk on sidewalks or crosswalks, and bicycles should drive in nonmotorized lanes. We consider this constraint as a loss function, we predict the absolute trajectory s_{abs} based on the definition of a constraint region $\mathcal{R}_t^{(i)}$, whose center is $(c_x^{(i)}, c_y^{(i)})$ and whose width and height are $w^{(i)}$ and $h^{(i)}$, respectively. For a prediction point $s_{abs,t}^{(i)} = (x_{t,n}, y_{t,n})$, we define its region loss as 0 if it is located within the constraint region; otherwise, we calculate its normalized distance to the constraint region and take the minimum value as the region loss of the point. The final region loss is the squared cumulative sum of the normalized distance over all time steps and objects, and it is calculated as follows:

$$d_{t,n} = \begin{cases} \left| x_{t,n} - \left(c_x^{(i)} - \frac{w}{2} \right) \right| & \text{if } x_{t,n} < c_x^{(i)} - \frac{w}{2} \\ \left| x_{t,n} - \left(c_x^{(i)} + \frac{w}{2} \right) \right| & \text{if } x_{t,n} > c_x^{(i)} + \frac{w}{2} \\ \left| y_{t,n} - \left(c_y^{(i)} - \frac{h}{2} \right) \right| & \text{if } y_{t,n} < c_y^{(i)} - \frac{h}{2} \\ \left| y_{t,n} - \left(c_y^{(i)} + \frac{h}{2} \right) \right| & \text{if } y_{t,n} > c_y^{(i)} + \frac{h}{2} \\ 0 & \text{if in region.} \end{cases} \quad (12)$$

$$\mathcal{L}_{\text{region}} = \frac{\varepsilon}{N * T} \sum_{t=1}^T \sum_{n=1}^N \text{ReLU} \left(\frac{d_{t,n}}{\max(\frac{w}{2}, \frac{h}{2})} \right)^2 \quad (13)$$

The model prediction accuracy is evaluated via the negative log-likelihood loss (NLL-loss) of the bivariate normal distribution.

$$\mathcal{L}_{\text{NLL}} = -\log p(\mathbf{x} | \mu, \Sigma) \quad (14)$$

where $\mathbf{x}$ is a two-dimensional data point, μ is the mean vector of the bivariate normal distribution and Σ is the covariance matrix.

For model complexity constraints, we synthesize a variety of computational neural network attributes and guide the TrajGNAS process by backpropagating the loss values induced during the supernet training process. We define the computational loss, which reflect the overall computational complexity of the model, focusing on the numbers of floats and parameters, which is given by:

$$\mathcal{L}_{\text{flops}} = \partial \cdot \left(\lambda_f \cdot \sum_{W_{\text{conv}} \in \mathcal{M}_{\text{conv}}} \mathbb{N}(W_{\text{conv}}) \cdot K_h \cdot K_w \cdot C_{\text{out}} + \lambda_f \cdot \sum_{W_{\text{fc}} \in \mathcal{M}_{\text{fc}}} \mathbb{N}(W_{\text{fc}}) \right) \quad (15)$$

Our final loss is a weighted average of all the above losses, and by introducing a multi-indicator loss function based on a posteriori knowledge and social constraints, the trajectory prediction model can achieve an optimal balance between accuracy and speed with interpretability. The total loss function is denoted as shown below:

$$\mathcal{L}_{\text{total}} = \varpi_1 \cdot \mathcal{L}_{\text{region}} + \varpi_2 \cdot \mathcal{L}_{\text{NLL}} + \varpi_3 \cdot \mathcal{L}_{\text{flops}} \quad (16)$$

where $\varpi_1, \varpi_2, \varpi_3$ are the definable weights for the different losses, and the user can adjust the values of these weights as needed to achieve a more flexible network architecture for searching.

4. Experiments

4.1. Datasets

In this work, we validate the effectiveness of the proposed method on two widely used and challenging public datasets: nuScenes [1] and Argoverse [3]. nuScenes is an autonomous driving dataset provided by Motional that contains multimodal sensor data such as surround view cameras, millimeter-wave radar, LiDAR and GPS coordinates. The dataset contains 1,000 different urban traffic scenes from Singapore and Boston, each of which is approximately 20 seconds long, totaling approximately 14,000 keyframes of data covering a wide range of weather, lighting, and traffic conditions. Argoverse contains both a trajectory prediction dataset and a perception dataset, where the trajectory prediction dataset consists of 324,000 annotated dynamic

target trajectories that record the past 2 seconds of motion history and are required to predict the trajectory of the target for the next 3 seconds.

4.2. Metrics

In this work, standard evaluation metrics are used to measure the achieved prediction performance, including the minimum average distance error (minADE), minimum final distance error (minFDE) and missing rate (MR). ADE is the minimum value of the Euclidean distance between the predicted trajectory Y_t and the ground-truth trajectory $\hat{Y}_t^k$ averaged over the prediction step T for k predicted trajectories. FDE is used to measure the error between the endpoints of the predicted trajectory and the true trajectory, refers to the ratio of Euclidean distances between the prediction and the true situation at the final time step T for all K predicted trajectories that are greater than $2m$, which reflects the ability of the model to cover the trajectory prediction task.

4.3. Overall Performance Evaluation

To verify the superiority of the proposed method, we compare it with many classic and state-of-the-art methods on the two datasets. The results show that the performance of the models obtained by our proposed TrajGNAS approach is comparable to that of the state-of-the-art models, as shown in Tables 2 and 3, where we indicate the best performance metrics in **bold**, and the second- and third-ranked results are underlined. To facilitate the comparison, we do not compare the model computations and parameter counts here but only the accuracy metrics presented in Section 4.2.

Method	mADE	mFDE
LaneGCN [23]	0.87	1.36
TNT [43]	0.91	1.45
NLNI[44]	0.79	1.25
THOMAS[12]	0.94	1.44
DenseTNT[15]	0.88	1.28
mmTrans [27]	0.84	1.34
GOHOME[13]	0.89	1.29
LTP[37]	0.83	1.29
HiVT[46]	0.77	1.17
MultiModalTrans[17]	0.84	1.29
TPCN[39]	0.82	1.24
FRM[32]	<u>0.82</u>	1.27
ProphNet[38]	0.77	<u>1.14</u>
LAformer[25]	0.77	<u>1.16</u>
SIMPL [42]	<u>0.80</u>	1.18
TrajGNAS(ours)	<u>0.82</u>	1.10

Table 2. Performance comparison of @3s predictions on Argoverse

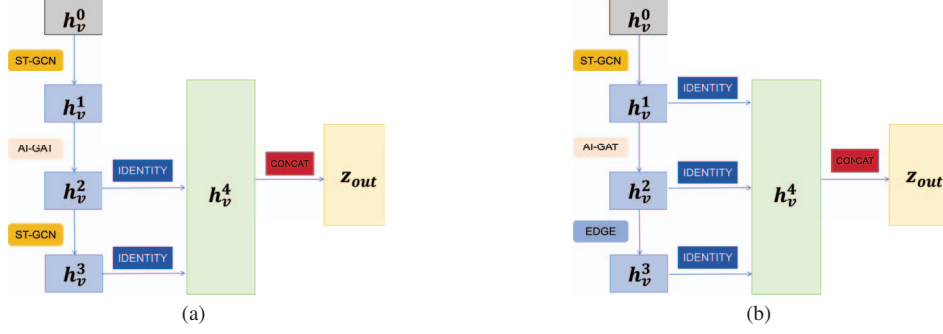


Figure 2. The searched architectures by TrajGNAS on different datasets. (a) Argoverse (b) nuScenes

4.3.1. Results Obtained on the Argoverse Dataset

Table 2 shows the results of the overall evaluation conducted on the Argoverse dataset when $K = 6$. Our proposed method ranks third in terms of the mADE metric but is the state-of-the-art approach in terms of the mFDE metric. In long-term prediction cases, which are reflected in the FDE metrics, our model incorporates semantic information and designs risk interaction mechanisms and LSTM networks with heterogeneous perceptions for different types of agents, which allows the model to consider factors over a longer time span when making predictions. The final architecture search result is shown in Fig. 2a.

4.3.2. Results Obtained on the nuScenes Dataset

The experimental results obtained on the nuScenes dataset show that our proposed method ranks among the top three in terms of the mADE metric when $K=10$ in Table 3. Notably, at both $K=5$ and $K=10$, the model achieves the best performance with respect to the MR evaluation metric, which corresponds to average improvements of 31.6% and 50% over DMAP (the second-place approach), respectively. This result indicates that our model is able to predict the final position of the agent more accurately. Our final architecture search result is shown in Fig. 2b.

4.4. Ablation Studies

To validate the effectiveness of our proposed method, we conduct a series of ablation experiments on multiple datasets to evaluate its innovative physics and risk interaction bootstrapping mechanism and the optimized multicriteria loss function in terms of the balance attained between accuracy and inference speed. In this section, we include both the number of parameters and the number of FLOPs as evaluation criteria. Tables 4 show the results of the ablation experiments performed on the nuScenes datasets. "GNAS" indicates whether a graph neural architecture search is used, "RI" indicates the presence or absence of a risk interaction mechanism, "Region-Loss" indicates whether the a posteriori region loss function is used, and "Compute-Loss" indicates whether model computational degree constraints are

Method	K=5		K=10	
	mADE	MR	mADE	MR
MultiPath[2]	2.32	-	1.96	-
Trajectron++[33]	1.88	0.7	1.51	0.57
P2T[5]	1.45	0.64	1.16	0.46
LaneGCN[23]	-	0.49	<u>0.95</u>	0.36
AgentFormer[40]	1.86	-	1.45	-
ALAN[28]	1.87	0.60	1.22	0.49
LaPred[20]	1.47	0.53	1.12	0.46
Autobot[14]	1.37	0.62	1.03	0.44
THOMAS[12]	1.33	0.55	1.04	-
GOHOME[13]	1.42	0.57	1.15	0.47
PGP[6]	1.30	0.61	1.00	0.37
Q-EANet[4]	<u>1.18</u>	<u>0.48</u>	1.02	0.44
DMAP[29]	1.09	<u>0.19</u>	1.07	<u>0.18</u>
LAFormer[26]	<u>1.19</u>	<u>0.48</u>	0.93	<u>0.33</u>
TrajGNAS (ours)	1.20	0.13	<u>0.99</u>	0.09

Table 3. Performance comparison of @6s predictions on nuScenes

used.

As shown in Table 4, our proposed risk interaction mechanism and regional semantic constraint loss significantly improve the accuracy of the model. In terms of accuracy metrics, the introduction of the risk interaction mechanism significantly improves the mADE, mFDE, and MR metrics of the model relative to those attained in the case where only the GNAS is used, as shown in Models 1 and 2. In addition, the adoption of regional semantic constraints for the losses also improves these three indicators, especially mFDE and MR, which are more significant. This indicates that the region semantic constraint has a good regulating effect on the task of predicting the final position of the agent. It is able to restrict the generated trajectories to semantically matched regions and eliminate the predicted trajectories that are out of the predefined range by imposing penalties.

The visualization results presented in Fig. 3 show how effectively the region semantic constraints constrain the trajectory points. As shown, without constraints, the trajectory

Model (ID)	GNAS	RI	Region-Loss	Compute-Loss	mADE	mFDE	MR	Params	FLOPS (MB)
1	✓				1.532	1.911	0.53	1089K	8.961
2	✓	✓			1.103	1.518	0.29	1315K	9.346
3	✓	✓	✓		0.990	1.312	0.09	1364K	9.764
4	✓	✓	✓	✓	0.997	1.326	0.09	1031K	8.716

Table 4. Results of ablation experiments conducted on nuScenes with K=10.

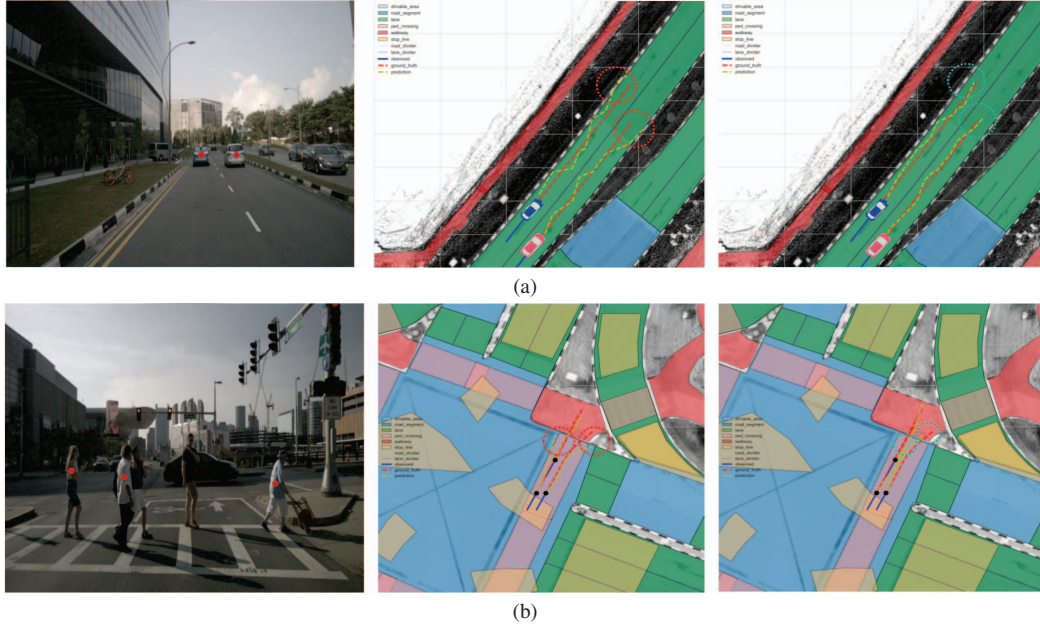


Figure 3. Qualitative results of two scenes in nuScenes dataset demonstrate the limiting effect of regional semantic loss on predicted trajectories. (a) two-vehicle side-by-side driving case; and (b) multiple pedestrians are walking on zebra region(predict three of them). The blue solid line in the figure indicates the observed trajectory, the red dashed line represents the ground truth, and the green dashed line represents the predicted trajectory. The red circle indicates a predicted trajectory that deviates from the corresponding road semantics, and the green circle indicates a predicted trajectory that is corrected via the region semantic loss.

ries predicted by the model tend to deviate far from the predicted endpoints (as shown by the red circles); these deviations are clearly unreasonable and can have a great effect on the subsequent decisions made by the self-driving vehicle, especially when the prediction task is performed over a long period of time. By imposing semantic constraints on road zones, the predicted endpoint location is limited to a reasonable range (as shown by the green circle), and the large deviation exhibited by the trajectory during the process is also significantly reduced. This is crucial for implementing safe and reliable decision-making in self-driving vehicles.

By introducing a computational loss function, TrajGNAS effectively guides the network search process, and the number of parameters and computational burden for the final searched model are reduced; however, the impact of this configuration on the accuracy is not significant. mADE and mFDE only increase by 7% and 10.6%, respectively, and

there is no degradation in the MR performance metric.

5. Conclusion

In this work, we firstly propose a NAS method, TrajGNAS, for trajectory prediction tasks; this approach can automatically search for the optimal GNN to construct a trajectory prediction model in a specified autonomous driving scenario to better model heterogeneous graphs. The search process takes into prediction accuracy, computational complexity, and interpretability of the model into account and finally finds the optimal structure that is suitable for the trajectory prediction task in the current scenario and the actual deployment requirements.

References

- [1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Gi-

- ancarlo Baldan, and Oscar Beijbom. nuscenes: A multi-modal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1, 6
- [2] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*, 2019. 7
- [3] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757, 2019. 1, 6
- [4] Jiuyu Chen, Zhongli Wang, Jian Wang, and Baigen Cai. Q-ea-net: Implicit social modeling for trajectory prediction via experience-anchored queries. *IET Intelligent Transport Systems*, 18(6):1004–1015, 2024. 7
- [5] Nachiket Deo and Mohan M Trivedi. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv preprint arXiv:2001.00735*, 2020. 7
- [6] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, pages 203–212. PMLR, 2022. 7
- [7] Yuhui Ding, Quanming Yao, Huan Zhao, and Tong Zhang. Diffmg: Differentiable meta graph search for heterogeneous graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 279–288, 2021. 3
- [8] Jianwu Fang, Chen Zhu, Pu Zhang, Hongkai Yu, and Jianru Xue. Heterogeneous trajectory forecasting via risk and scene graph learning. *IEEE Transactions on Intelligent Transportation Systems*, 24(11):12078–12091, 2023. 1
- [9] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11525–11533, 2020. 2
- [10] Xing Gao, Xiaogang Jia, Yikang Li, and Hongkai Xiong. Dynamic scenario representation learning for motion forecasting with heterogeneous graph convolutional recurrent networks. *IEEE Robotics and Automation Letters*, 8(5):2946–2953, 2023. 2
- [11] Yang Gao, Hong Yang, Peng Zhang, Chuan Zhou, and Yue Hu. Graphnas: Graph neural architecture search with reinforcement learning. *arXiv preprint arXiv:1904.09981*, 2019. 2, 3
- [12] Thomas Gilles, Stefano Sabatini, Dzmityr Tsishkou, Bogdan Stanculescu, and Fabien Moutarde. Thomas: Trajectory heatmap output with learned multi-agent sampling. *arXiv preprint arXiv:2110.06607*, 2021. 6, 7
- [13] Thomas Gilles, Stefano Sabatini, Dzmityr Tsishkou, Bogdan Stanculescu, and Fabien Moutarde. Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 international conference on robotics and automation (ICRA)*, pages 9107–9114. IEEE, 2022. 6, 7
- [14] Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D’Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. *arXiv preprint arXiv:2104.00563*, 2021. 7
- [15] Junru Gu, Chen Sun, and Hang Zhao. Densentn: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021. 6
- [16] Zhao Huan, YAO Quanming, and TU Weiwei. Search to aggregate neighborhood for graph neural network. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pages 552–563. IEEE, 2021. 3
- [17] Zhiyu Huang, Xiaoyu Mo, and Chen Lv. Multi-modal motion prediction with transformer-based neural network for autonomous driving. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2605–2611. IEEE, 2022. 6
- [18] Xiaosong Jia, Liting Sun, Hang Zhao, Masayoshi Tomizuka, and Wei Zhan. Multi-agent trajectory prediction by combining egocentric and allocentric views. In *Conference on Robot Learning*, pages 1434–1443. PMLR, 2022. 1
- [19] Younkyung Jwa, Chang Wook Ahn, and Man-Je Kim. Egnas: Efficient graph neural architecture search through evolutionary algorithm. *Mathematics*, 12(23):3828, 2024.
- [20] ByeoungDo Kim, Seong Hyeon Park, Seokhwan Lee, Elbek Khoshimjonov, Dongsuk Kum, Junsoo Kim, Jeong Soo Kim, and Jun Won Choi. Lapred: Lane-aware prediction of multi-modal future trajectories of dynamic agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14636–14645, 2021. 7
- [21] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 2
- [22] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. Grip++: Enhanced graph-based interaction-aware trajectory prediction for autonomous driving. *arXiv preprint arXiv:1907.07792*, 2019. 1
- [23] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. 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*, pages 541–556. Springer, 2020. 2, 6, 7
- [24] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. 5
- [25] Mengmeng Liu, Hao Cheng, Lin Chen, Hellward Broszio, Jiangtao Li, Runjiang Zhao, Monika Sester, and Michael Ying Yang. Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2039–2049, 2024. 6
- [26] Mengmeng Liu, Hao Cheng, Lin Chen, Hellward Broszio, Jiangtao Li, Runjiang Zhao, Monika Sester, and Michael Ying Yang. Laformer: Trajectory prediction for autonomous driving with lane-aware scene constraints.

- In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2039–2049, 2024. 7
- [27] Yicheng Liu, Jinghui Zhang, Liangji Fang, Qinhong Jiang, and Bolei Zhou. Multimodal motion prediction with stacked transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7577–7586, 2021. 6
- [28] Sriram Narayanan, Ramin Moslemi, Francesco Pittaluga, Buyu Liu, and Manmohan Chandraker. Divide-and-conquer for lane-aware diverse trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15799–15808, 2021. 7
- [29] Alexander Naumann, Felix Hertlein, Daniel Grimm, Maximilian Zipfl, Steffen Thoma, Achim Rettinger, Lavdim Halilaj, Juergen Luetttin, Stefan Schmid, and Holger Caesar. Lanelet2 for nuscenes: Enabling spatial semantic relationships and diverse map-based anchor paths. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3247–3256, 2023. 7
- [30] Babatounde Moctard Oloulade, Jianliang Gao, Jiamin Chen, Tengfei Lyu, and Raed Al-Sabri. Graph neural architecture search: A survey. *Tsinghua Science and Technology*, 27(4): 692–708, 2021. 3
- [31] Zheyi Pan, Songyu Ke, Xiaodu Yang, Yuxuan Liang, Yong Yu, Junbo Zhang, and Yu Zheng. Autostg: Neural architecture search for predictions of spatio-temporal graph. In *Proceedings of the Web Conference 2021*, pages 1846–1855, 2021. 3
- [32] Daehee Park, Hobin Ryu, Yunseo Yang, Jegyeong Cho, Jiwon Kim, and Kuk-Jin Yoon. Leveraging future relationship reasoning for vehicle trajectory prediction. *arXiv preprint arXiv:2305.14715*, 2023. 6
- [33] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. 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*, pages 683–700. Springer, 2020. 1, 2, 7
- [34] Pei Sun, Henrik Kretschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 1
- [35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 2
- [36] Chalavadi Vishnu, Vineel Abhinav, Debaditya Roy, C Krishna Mohan, and Ch Sobhan Babu. Improving multi-agent trajectory prediction using traffic states on interactive driving scenarios. *IEEE Robotics and Automation Letters*, 8(5): 2708–2715, 2023. 1
- [37] Jingke Wang, Tengju Ye, Ziqing Gu, and Junbo Chen. Ltp: Lane-based trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17134–17142, 2022. 6
- [38] Xishun Wang, Tong Su, Fang Da, and Xiaodong Yang. Prophnet: Efficient agent-centric motion forecasting with anchor-informed proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21995–22003, 2023. 6
- [39] Maosheng Ye, Tongyi Cao, and Qifeng Chen. Tpcn: Temporal point cloud networks for motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11318–11327, 2021. 6
- [40] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823, 2021. 7
- [41] Kunpeng Zhang, Xiaoliang Feng, Lan Wu, and Zhengbing He. Trajectory prediction for autonomous driving using spatial-temporal graph attention transformer. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):22343–22353, 2022. 1
- [42] Lu Zhang, Peiliang Li, Sikang Liu, and Shaojie Shen. Simpl: A simple and efficient multi-agent motion prediction baseline for autonomous driving. *IEEE Robotics and Automation Letters*, 2024. 6
- [43] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Ben Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, et al. Tnt: Target-driven trajectory prediction. In *Conference on Robot Learning*, pages 895–904. PMLR, 2021. 6
- [44] Fang Zheng, Le Wang, Sanping Zhou, Wei Tang, Zhenxing Niu, Nanning Zheng, and Gang Hua. Unlimited neighborhood interaction for heterogeneous trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13168–13177, 2021. 6
- [45] Kaixiong Zhou, Xiao Huang, Qingquan Song, Rui Chen, and Xia Hu. Auto-gnn: Neural architecture search of graph neural networks. *Frontiers in big Data*, 5:1029307, 2022. 3
- [46] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8823–8833, 2022. 6