vehicles

MDPI

Article

PHR-Net: Proposal-Level Historical Retrieval for Non-Stationary Temporal Consistency in Trajectory Prediction

Bo Zhang and Ming Xu *

School of Software, Liaoning Technical University, Huludao 125105, China; zhangbo15931402792@163.com
* Correspondence: xum.2016@tsinghua.org.cn

Abstract

Multi-agent trajectory prediction serves as a critical component in autonomous driving systems, bridging environment perception, behavior understanding, and motion planning. Its outputs not only affect candidate trajectory evaluation and interactive decision-making but also directly influence downstream processes such as risk anticipation, braking and yielding, and safety margin allocation. Therefore, obtaining accurate and stable prediction results is of great importance. Although existing methods have achieved remarkable progress in single-timestep prediction accuracy, most of them still adopt an independent decoding paradigm under a sliding-window setting. As a result, during continuous online prediction, these models are prone to frequent mode switching, temporal discontinuities in overlapping segments, and local trajectory jitter, which become particularly pronounced in complex interactive scenarios such as yielding, merging, and unprotected turning. To address these issues, this paper proposes PHR-Net, a two-stage proposal-level historical retrieval framework that introduces cross-timestep historical context to perform consistency-aware refinement of current predictions on top of multimodal coarse proposals. Experiments on the Argoverse 1 benchmark show that PHR-Net achieves competitive performance under both Top-1 and Top-6 settings. PHR-Net obtains a Top-1 minFDE of 1.0834 and MR of 0.1046 and achieves an MR of 0.1027 under the Top-6 setting. In the overlapping-interval consistency evaluation, PHR-Net reduces the summed ADE to 2.08. These results show that proposal-level historical retrieval improves endpoint reliability and cross-timestep temporal consistency.

Keywords: trajectory prediction; autonomous driving; temporal consistency; historical retrieval; multimodal decoding; decoupled modeling

Academic Editors: Muhammad Hussain, Xiaomeng Li, Paul Roberts and Amjad Pervez

Received: 30 March 2026
Revised: 29 April 2026
Accepted: 11 May 2026
Published: 12 May 2026

1. Introduction

Trajectory prediction is a critical intermediate component in autonomous driving systems, connecting perception, interaction understanding, and planning and decision-making. Its task is to predict the distribution of possible future trajectories over the next several seconds, given historical observations, surrounding agents, and high-definition maps. Traditional approaches typically rely on Kalman filtering, social force models, or handcrafted prior rules. While these methods offer good interpretability in simple traffic environments, they often struggle to adequately capture behavioral diversity and long-term uncertainty in complex urban roads, dense intersections, and multi-agent interaction scenarios [1,2].

With the development of deep learning, the modeling paradigm for trajectory prediction has evolved substantially. Early studies primarily focused on sequential modeling

and social interaction. Representative works, such as Social LSTM, DESIRE, Social GAN, Trajectron, and Trajectron++, improved multimodal representation from the perspectives of social pooling, latent-variable sampling, adversarial multimodal generation, dynamic graph modeling, and dynamic feasibility constraints, respectively [3–7]. Subsequently, research attention gradually shifted toward representations better suited to autonomous driving scenarios, moving from image rasterization to vectorized maps, graph neural networks, and Transformer-based architectures. Representative methods, including VectorNet, LaneGCN, HiVT, and HDGT, emphasize geometric fidelity, lane topology, hierarchical attention, and heterogeneous graph fusion, respectively [8–11]. In recent years, multimodal prediction has been further formulated as a structured query process or a generative modeling problem. Typical methods include MultiPath, CoverNet, TNT, MTR, QCNet, Wayformer, and MotionDiffuser, corresponding to different paradigms such as anchor-based classification, target-point-driven prediction, query-based coarse-to-fine decoding, unified attention backbones, and diffusion-based generation [12–17].

However, most of the above methods still treat "one-shot prediction at the current timestep" as the basic unit of operation. In online systems, the observation window slides from time step t to $t + \Delta t$, and the two windows share highly overlapping historical inputs. Correspondingly, the two future predictions also overlap substantially on the real temporal axis. If the model produces conflicting trajectories within these overlapping intervals, abruptly switches high-level modes, or yields terminal predictions with large directional oscillations at adjacent timesteps, the downstream planner may be forced to respond excessively to prediction noise. In other words, low average error under static evaluation does not necessarily imply sufficient temporal stability in streaming deployment.

Recent studies on dynamic prediction have begun to directly address this issue. An early representative work, DCMS, approached the problem from the perspective of online streaming deployment and pointed out that consecutive predictions in autonomous driving systems are not independent of one another but should instead satisfy cross-timestep temporal continuity and spatial robustness. To this end, it introduced temporal and spatial consistency constraints during training, encouraging outputs from adjacent temporal windows to remain as consistent as possible within overlapping intervals, thereby reducing prediction jitter and improving stability in continuous deployment [18]. Building on this line of work, HPNet further showed that the relationship between predictions at adjacent timesteps cannot always be simplified as strict consistency over overlapping segments. In complex interactive scenarios, two consecutive predictions may differ locally while still sharing the same high-level motion objective. This suggests that cross-timestep relationships are essentially better characterized as a more general form of dynamic correlation rather than forced alignment [19]. Zang et al. proposed a framework consisting of motion-primitive-based reference path planning and dynamic-programming-based spatio-temporal trajectory planning, aiming to generate safe, smooth, and dynamically feasible trajectories for AGVs in off-road environments [20]. This further indicates that autonomous driving systems require stable and smooth trajectory information during continuous operation.

Based on this observation, this paper proposes PHR-Net. The proposed framework retains the two-stage coarse-to-fine decoding paradigm of QCNet [16] while redefining the role of the second stage. In the first stage, coarse proposals are generated in parallel. In the second stage, proposals are treated as slot anchors, and historical consistency is decomposed into two complementary levels: high-level intent stability and local trajectory geometric alignment. The former is used to reduce mode switching, while the latter improves continuity over overlapping temporal intervals. To enable the two branches to share an explicit geometric reference, the current proposals, historical proposals, and memory prototypes are further unified within the same agent-centric local frame.

The main contributions of this paper can be summarized as follows. First, we propose a proposal-level historical retrieval framework for non-stationary temporal consistency, which explicitly models historical information at the coarse proposal level while preserving parallel training. Second, we design a dual-branch refinement strategy: the intent branch is responsible for high-level mode stability, while the Trajectory Branch focuses on local geometric continuity, and both branches aggregate historical information under a unified slot-based formulation.

The remainder of this paper is organized as follows. Section 2 reviews related work on classical trajectory modeling, vectorized scene representation, multimodal prediction, query-based modeling, and non-stationary temporal consistency. Section 3 presents the proposed PHR-Net framework, including the two-stage decoding architecture, intention branch, Trajectory Branch, residual refinement, and training objective. Section 4 describes the experimental setup and analyzes the quantitative results, qualitative visualizations, ablation studies, and temporal consistency evaluation. Section 5 concludes the paper and discusses future research directions.

2. Related Work

This section reviews the research most relevant to the proposed PHR-Net. We first summarize classical trajectory modeling and interaction learning methods and then discuss vectorized scene representation, multimodal prediction, and query-based modeling. Finally, we review recent studies on temporal consistency in continuous prediction, which are closely related to the motivation of this work.

2.1. Classical Trajectory Modeling and Interaction Learning

Early trajectory prediction research primarily focused on single-agent motion modeling and explicit interaction modeling. Kalman filtering and social force models represent two classical paradigms, namely linear dynamics and crowd-behavior-prior-based modeling, respectively [1,2]. In the deep learning era, Social LSTM was the first to introduce social pooling of surrounding pedestrians into sequence prediction tasks [3]. DESIRE modeled future uncertainty through conditional variational inference [4]. Social GAN further generated diverse trajectories by leveraging adversarial learning and the variety loss [5]. The Trajectron and Trajectron++ emphasized dynamic graph-based interaction modeling and dynamical feasibility constraints, respectively [6,7]. The value of this line of research lies in revealing the importance of interaction modeling and multimodal modeling. However, its input representations and output structures remain oriented toward general scenarios and have not yet fully exploited map semantics, lane topology, and long-horizon goal constraints in autonomous driving.

2.2. Vectorized Scene Representation

Another core issue in autonomous driving trajectory prediction is how to represent scene context. Early raster-based methods typically rendered historical trajectories, lane boundaries, and traffic elements into multi-channel images, which were then encoded by CNNs. Representative works include IntentNet and Rules of the Road [21,22]. Although raster representations facilitate the reuse of visual backbones, they also introduce rendering loss, memory overhead, and resolution dependence. To improve efficiency and geometric fidelity, the research community gradually shifted toward vectorized representations. VectorNet unified the modeling of lane and trajectory information through polyline structures, initiating the mainstream direction of vectorized representation [8]. LaneGCN explicitly exploited lane graphs to preserve topological connectivity and actor–lane interactions [9]. HiVT and HDGT further incorporated local geometric normalization, hierarchical

graph structures, and Transformers into a unified framework, improving efficiency and generalization in large-scale scenarios [10,11].

2.3. Multimodal Outputs

Multimodal future distributions are the standard output form in autonomous driving prediction, but how to learn multiple plausible modes in a stable manner remains a persistent challenge. One typical direction is the anchor-based paradigm. MultiPath represents discrete modes using fixed trajectory anchors [12]. CoverNet formulates the problem as trajectory-set classification to avoid mode averaging caused by direct regression [13]. TNT further regards endpoints as the primary source of uncertainty, first selecting target points and then completing the full trajectory [14]. Such methods usually offer good interpretability, but their performance is still affected by anchor coverage, candidate density, and post-processing strategies. Another line of work has begun to de-emphasize explicit anchor design and instead adopt direct generative or anchor-free multimodal decoding. Scene Transformer, from the perspective of unified scene modeling, formulates multi-agent futures as a conditional sequence generation problem [23]. AgentFormer directly models the joint future distribution through cross-agent and cross-temporal attention [24]. Wayformer further employs a more homogeneous attention backbone and learnable queries to directly produce multimodal outputs [25]. This line of research no longer relies on predefined trajectory sets or target-point candidates but instead allows the model to directly learn the correspondence between mode slots and future trajectories. To combine the advantages of both paradigms, MTR uses learnable motion query pairs to decouple global intention localization from local motion refinement [15], while QCNet adopts query-centric encoding together with proposal-and-refinement design, achieving strong performance in both online inference efficiency and long-horizon prediction [16]. An increasing number of studies indicate that the coarse-to-fine paradigm is an effective way to balance training stability and multimodal coverage. This paper inherits this overall structure but advances the core of the second stage from static context refinement to proposal-level historical retrieval.

2.4. Query-Based Modeling

At the level of scene fusion and decoder design, the Transformer has become the dominant backbone. Scene Transformer treats multi-agent futures as conditional sequence outputs from the perspective of unified scene modeling [23]. AgentFormer emphasizes joint attention across both agents and time [24]. Wayformer demonstrates that a more homogeneous stack of attention layers can also achieve strong performance on large-scale benchmarks [25]. In parallel, M2I and MotionDiffuser extend the capability boundary of query-based predictors from the perspectives of interactive joint prediction and generative sampling, respectively [17,26]. Collectively, these works have promoted a common trend: organizing multimodal futures as a set of structured query slots, rather than treating multiple future trajectories as merely auxiliary end outputs. Our method also follows this idea. However, our focus is not merely on generating better slots but on how the same slot can maintain an interpretable historical continuity relationship across consecutive timesteps.

2.5. Non-Stationary Temporal Consistency

Compared with one-shot prediction under static windows, online deployment is more concerned with whether outputs at adjacent timesteps remain coherent. From an earlier perspective, recursive state estimation methods such as Kalman filtering essentially perform continuous state updates along the temporal axis [1], while dynamical methods such as social force models also attempt to describe agent motion and interactions through continuous evolution equations [2]. Subsequently, deep learning methods extended this notion of temporal evolution to more complex multi-agent scenarios. The Trajectron modeled

continuous interaction processes through dynamic graphs and temporal latent states [6], while MANTRA and R2P2 enhanced future evolution modeling using external memory and generative dynamics, respectively [27,28]. However, these methods still mainly focused on improving the dynamic expressiveness of single-shot prediction, rather than explicitly constraining the consistency of adjacent predictions under sliding windows over overlapping time intervals. To achieve more stable prediction, the representative method DCMS highlighted that consecutive prediction results should simultaneously satisfy temporal continuity and spatial stability. It therefore introduced dual consistency constraints during training, suppressing prediction jitter through time-shifted overlapping windows and spatial perturbation augmentation [18]. Subsequently, HPNet further observed that the relationship between predictions at adjacent timesteps is not necessarily equivalent to pointwise alignment. In complex interactive scenarios, they may differ locally while still sharing a consistent high-level motion objective [19]. This indicates that cross-timestep temporal consistency exhibits a non-stationary nature. The proposal-level historical retrieval design in this paper is developed within this line of research but further shifts the focus down to the historical continuity of the coarse proposal slots themselves.

3. Method

This section first presents the problem formulation and the overall architecture, and then describes the three types of backbone attention, the two-stage proposal-level historical retrieval mechanism, and the training objectives in sequence. Unless otherwise specified, all positions, displacements, and trajectories are represented in the agent-centric local frame of the target agent at the current timestep.

3.1. Problem Formulation

For the target agent i, the historical state sequence is denoted as $X_i = \{x_i^1, x_i^2, \ldots, x_i^{T_{\text{obs}}}\}$, where T_{obs} is the length of the historical observation window, and x_i^t contains attributes such as position, velocity, heading, and agent type. The neighboring agent set is denoted as $N_i = \{X_j \mid j \in \Omega_i\}$, where Ω_i represents the index set of neighboring agents that may interact with the target agent. The local map token set is denoted as $M_i = \{m_i^1, m_i^2, \ldots, m_i^{L_m}\}$, where m_i^l represents the l-th map semantic unit. Each map semantic unit corresponds to semantic information such as lane centerline segments, boundary segments, and intersection regions. Let T_{fut} denote the future prediction horizon, and let K denote the number of candidate motion modes generated by the model. Then, the future trajectory of agent i under the k-th mode and the probability distribution over all modes can be represented as

$$Y_i^k = \{y_i^{k,1}, y_i^{k,2}, \ldots, y_i^{k,T_{\text{fut}}}\}, \quad \Pi_i = \{\pi_i^1, \pi_i^2, \ldots, \pi_i^K\}, \quad \sum_{k=1}^{K} \pi_i^k = 1. \tag{1}$$

Here, $y_i^{k,\tau}$ denotes the predicted state of agent i at the future time step τ under the k-th mode, where $\tau \in \{1, 2, \ldots, T_{\text{fut}}\}$. The term π_i^k denotes the probability of the k-th predicted mode.

3.2. Overall Architecture

As shown in Figure 1, PHR-Net adopts a two-stage coarse-to-fine prediction architecture for continuous trajectory prediction. The first stage is responsible for generating diverse coarse proposals from the current observation window, while the second stage introduces proposal-level historical retrieval to refine these proposals in a consistency-aware manner.

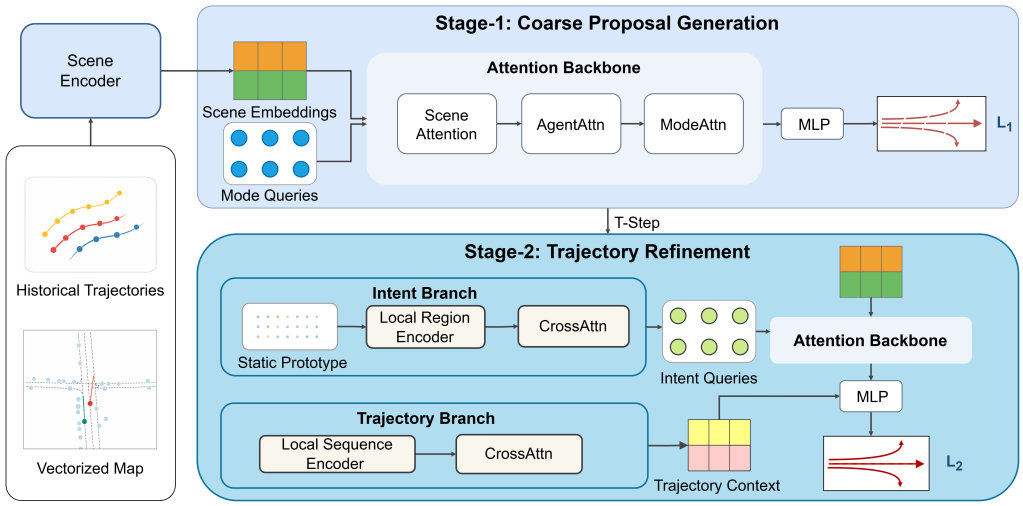

Figure 1. Overall architecture of PHR-Net. Stage-1 generates multimodal coarse proposals from the current sliding window. These proposals are stored as historical memory and used by Stage-2 for intention-history retrieval and trajectory-history retrieval. The retrieved historical context is fused with the current mode features to generate residual corrections and final refined trajectories.

In Stage-1, the model first encodes the target-agent historical trajectory X_i, neighboring-agent set N_i, and local map elements M_i into graph-based node and edge representations. These representations are then processed by scene attention, agent attention, and mode attention in sequence. The scene attention module introduces historical and map context into each motion query. The agent attention module aggregates interaction information from neighboring agents. The mode attention module further coordinates the K candidate motion hypotheses of the target agent and reduces redundant modes. After these attention operations, each mode-specific representation $z_{i,k}^{m}$ is decoded by a lightweight MLP into a coarse trajectory proposal $L_{i,k}^{1}$. In Stage-2, PHR-Net no longer reconstructs the future trajectory from scratch. Instead, it treats the Stage-1 coarse proposals $L_{i,k}^{1}$ as slot-level anchors and retrieves relevant historical information from the historical coarse proposal memory. The refinement process is decomposed into two complementary branches. The intention branch extracts a compact high-level behavior representation $g_{i,k}$ from each current coarse proposal $L_{i,k}^{1}$, matches it with the static prototype library P, and retrieves historical intention information to maintain high-level mode stability. The retrieved intention representation is then used as the initial query of the second-stage backbone and is further updated together with the current scene representation through scene attention, agent attention, and mode attention, thereby obtaining the intention-aware second-stage mode feature $z_{i,k}^{m,2}$. Meanwhile, the Trajectory Branch directly encodes current and historical proposal sequences and performs relation-aware alignment over historical proposals to retrieve local geometric references from overlapping temporal intervals, yielding the trajectory-history feature $z_{i,k}^{H}$. Finally, the second-stage mode feature $z_{i,k}^{m,2}$ is fused with the trajectory branch output $z_{i,k}^{H}$ to predict the residual correction $\Delta L_{i,k}$, and the final trajectory is obtained as $L_{i,k}^{2} = L_{i,k}^{1} + \Delta L_{i,k}$.

The parallel decoding mentioned in this paper means that, during training, the coarse proposals $L_{i,k}^{1}$ can be constructed from shifted observation windows within the same batch, because Stage-1 does not introduce any historical information. In Stage-2, historical proposals are used as memory tokens, and the retrieval weights over the retained historical windows and proposal tokens are computed through parallel attention operations. During online inference, as new sliding windows arrive, the proposal memory is updated chronologically. This indeed introduces a natural dependency on previous prediction windows, which is unavoidable in streaming deployment. Appendix A provides a detailed algo-

rithm flowchart in Figure A1 and the correspondence between compact and full notation in Table A1.

3.3. Scene Attention

All inputs of the model are first represented as nodes or edges on a graph and then encoded into hidden features. For the i-th agent node, we preserve attributes that are directly related to its motion state, including position, velocity, heading angle, and category information, and organize them as the raw agent attribute vector x_i^a. It can be written as $x_i^a = (p_i, v_i, \theta_i, c_i)$, where p_i denotes the position, v_i denotes the velocity, θ_i denotes the heading angle, and c_i denotes the category label. Both the target agent and neighboring agents are encoded by a two-layer MLP, obtaining the agent node feature $h_i^a = \mathrm{MLP}(x_i^a)$. The encoded h_i^a is used as the agent node representation for subsequent interaction modeling among different traffic participants. For the j-th map node, its geometric and semantic attributes are described. Each map node is constructed from a lane semantic unit, and its attributes include length, intersection indicator, turning attribute, and traffic control state. These attributes are organized as the raw map attribute vector $x_j^m = (l_j, I_j, d_j, s_j)$, where l_j denotes the length, I_j denotes the intersection indicator, d_j denotes the turning attribute, and s_j denotes the traffic control attribute. After being encoded by a two-layer MLP, the map node feature is obtained as $h_j^m = \mathrm{MLP}(x_j^m)$. Edges are used to describe the relative relationships between nodes. For an arbitrary source node and target node, the edge attributes include the distance d_{ij} from the source node to the target node, the edge direction φ_{ij} in the coordinate frame of the target node, the relative heading difference ψ_{ij} between the source node and the target node, and the temporal difference δ_{ij} between them. Therefore, the edge attribute vector and its encoded representation are defined as $r_{ij} = (d_{ij}, \varphi_{ij}, \psi_{ij}, \delta_{ij})$ and $e_{ij} = \mathrm{MLP}(r_{ij})$, respectively. Let $q_{i,k}^0$ denote the initial query vector corresponding to the k-th mode. The scene-conditioned attention aggregates context from both historical agent nodes and map nodes and can therefore be formulated as relation-aware cross-attention. Let C_i denote the set of scene context nodes for agent i, where c_j is the feature of the j-th context node. The query, key, and value are defined as

$$q_{i,k}^s = W_Q^s q_{i,k}^0, \tag{2}$$

$$k_{i,j}^s = W_K^s c_j + W_{Ke}^s e_{i,j}, \tag{3}$$

$$v_{i,j}^s = W_V^s c_j + W_{Ve}^s e_{i,j}. \tag{4}$$

Here, W_Q^s, W_K^s, and W_V^s denote the query, key, and value projection matrices, respectively, while W_{Ke}^s and W_{Ve}^s are used to project the edge feature $e_{i,j}$ into the key and value spaces. The scene attention weight and the corresponding output are further computed as

$$\alpha_{i,j,k}^s = \mathrm{softmax}_{j \in C_i}\left(\frac{{q_{i,k}^s}^T k_{i,j}^s}{\sqrt{d}}\right), \tag{5}$$

$$z_{i,k}^s = \sum_{j \in C_i} \alpha_{i,j,k}^s v_{i,j}^s. \tag{6}$$

After this step, $z_{i,k}^s$ has integrated the spatiotemporal semantic information of the current agent in the local scene.

3.4. Agent Attention

In real traffic scenarios, the future motion of an agent is not only influenced by road topology and historical states but also strongly dependent on the interactive behaviors of neighboring agents. Therefore, after obtaining the scene features, it is necessary to further

model the dynamic interaction relationships between the target agent and its neighboring agents. The query is derived from the output $z^s_{i,k}$ of the previous scene-conditioned attention module, and the query, key, and value are defined as

$$q^a_{i,k} = W^a_Q z^s_{i,k}, \tag{7}$$

$$k^a_{i,j} = W^a_K h^a_j + W^a_{Ke} e_{i,j}, \tag{8}$$

$$v^a_{i,j} = W^a_V h^a_j + W^a_{Ve} e_{i,j}. \tag{9}$$

Here, W^a_Q, W^a_K, and W^a_V denote the query, key, and value projection matrices in the agent attention module, respectively, while W^a_{Ke} and W^a_{Ve} are used to project the edge feature $e_{i,j}$ into the key and value spaces. The agent-interaction attention weight and the corresponding output are further computed as

$$\alpha^a_{i,j,k} = \mathrm{softmax}_{j \in N_i}\left(\frac{{q^a_{i,k}}^T k^a_{i,j}}{\sqrt{d}}\right), \tag{10}$$

$$z^a_{i,k} = \sum_{j \in N_i} \alpha^a_{i,j,k} v^a_{i,j}. \tag{11}$$

Through this module, $z^a_{i,k}$ not only preserves the agent's own scene information but also further incorporates the potential influence of neighboring agents on its future motion at the same time step.

3.5. Mode Attention

Mode attention is used to coordinate multiple future hypotheses within the same agent, reduce redundant modes, and enhance the complementarity among different modes. In this stage, the relative relationships of edges are no longer used, and self-attention is performed only along the mode dimension inside the same target agent. For clarity, the query of mode k and the key and value of mode k' are defined as

$$q^m_{i,k} = W^m_Q z^a_{i,k}, \tag{12}$$

$$k^m_{i,k'} = W^m_K z^a_{i,k'}, \tag{13}$$

$$v^m_{i,k'} = W^m_V z^a_{i,k'}. \tag{14}$$

Here, W^m_Q, W^m_K, and W^m_V denote the query, key, and value projection matrices in the mode-interaction attention module, respectively. The mode-interaction attention weight and the corresponding output are further computed as

$$\beta^m_{i,k,k'} = \mathrm{softmax}_{k'}\left(\frac{{q^m_{i,k}}^T k^m_{i,k'}}{\sqrt{d}}\right), \tag{15}$$

$$z^m_{i,k} = \sum_{k'=1}^{K} \beta^m_{i,k,k'} v^m_{i,k'}. \tag{16}$$

After obtaining the final mode representation $z^m_{i,k}$, an MLP is used to decode each mode in parallel, producing the k-th multimodal coarse proposal trajectory in the first stage:

$$L^1_{i,k} = \mathrm{MLP}(z^m_{i,k}). \tag{17}$$

The role of the coarse proposal is to provide a clear mode anchor for the second stage and serve as a historical reference, rather than directly regressing the final trajectory in a single step.

3.6. Stage-2: Intention Branch

The goal of the intention branch is to maintain the consistency of high-level behavioral patterns across consecutive predictions. Different from point-wise trajectory alignment, this branch focuses more on whether high-level behavioral modes such as going straight, turning left, merging, and decelerating remain stable along the continuous time axis. To this end, a compact high-level behavior representation is first extracted from the current coarse proposal, which can be defined as

$$g_{i,k} = \mathrm{MLP}(\mathrm{Pool}(L^1_{i,k}), p^e_{i,k}, u^e_{i,k}, \bar{v}_{i,k}). \tag{18}$$

Here, $\mathrm{Pool}(\cdot)$ denotes the statistical aggregation over the entire coarse proposal, $p^e_{i,k}$ denotes the trajectory endpoint, $u^e_{i,k}$ denotes the terminal displacement direction, and $\bar{v}_{i,k}$ denotes the average velocity of the whole proposal. Then, the current high-level behavior representation is matched with the static high-level mode prototype library $P = \{p_r\}_{r=1}^R$, and the weighted static prototype representation is obtained as

$$\beta^I_{i,k,r} = \mathrm{softmax}_r(g^T_{i,k} p_r), \tag{19}$$

$$p_{i,k} = \sum_{r=1}^{R} \beta^I_{i,k,r} p_r, \tag{20}$$

$$u^I_{i,k} = \mathrm{MLP}(g_{i,k}, p_{i,k}). \tag{21}$$

The relative relationship between the current query and the historical reference is still represented by a unified embedding. Specifically, it is defined as

$$r^I_{i,k,h} = (d^I_{i,k,h}, \varphi^I_{i,k,h}, \psi^I_{i,k,h}, \delta^I_{i,k,h}), \tag{22}$$

$$e^I_{i,k,h} = \mathrm{MLP}(r^I_{i,k,h}). \tag{23}$$

The historical high-level intention representation is obtained using the same aggregation strategy as the current proposal and is denoted as $g_{i,h}$. Then, the query, key, and value of the intention branch are defined as

$$q^I_{i,k} = W^I_Q u^I_{i,k}, \tag{24}$$

$$k^I_{i,k,h} = W^I_K g_{i,h} + W^I_{Ke} e^I_{i,k,h}, \tag{25}$$

$$v^I_{i,k,h} = W^I_V g_{i,h} + W^I_{Ve} e^I_{i,k,h}. \tag{26}$$

The intention-branch attention weight and the corresponding historical intention reference are further computed as

$$\alpha^I_{i,k,h} = \mathrm{softmax}_h\left(\frac{{q^I_{i,k}}^T k^I_{i,k,h}}{\sqrt{d}}\right), \tag{27}$$

$$z^I_{i,k} = \sum_{h=1}^{H} \alpha^I_{i,k,h} v^I_{i,k,h}. \tag{28}$$

Here, $z^I_{i,k}$ is used to represent the historical reference result of the current k-th mode at the high-level behavior level.

3.7. Stage-2: Trajectory Branch

The local trajectory branch directly models the correspondence between the current coarse proposal and historical coarse proposals at the local geometric level. Let $l^1_{i,k,\tau}$ denote the position of the current coarse proposal of agent i at the τ-th future time step, and let $\bar{l}^1_{i,h,t}$ denote the position of the historical coarse proposal from the h-th historical window at the t-th future time step. Then, the local sequential features of the current proposal and the historical proposal are encoded by a GRU as follows:

$$s_{i,k,\tau} = \mathrm{GRU}\left(l^1_{i,k,\tau}, \Delta l^1_{i,k,\tau}, \frac{\tau}{T_f}\right), \tag{29}$$

$$s_{i,h,t} = \mathrm{GRU}\left(\bar{l}^1_{i,h,t}, \Delta \bar{l}^1_{i,h,t}, \frac{t}{T_f}\right). \tag{30}$$

Here, $\Delta l^1_{i,k,\tau}$ and $\Delta \bar{l}^1_{i,h,t}$ denote the displacement increments between adjacent time steps, respectively, while τ/T_f and t/T_f are used to explicitly encode the relative temporal positions. To align the overlapping intervals between the current window and the historical window on the real time axis, a relation-aware alignment embedding is introduced as

$$r^H_{i,k,h,\tau,t} = (d^H_{i,k,h,\tau,t}, \varphi^H_{i,k,h,\tau,t}, \psi^H_{i,k,h,\tau,t}, \delta^H_{i,k,h,\tau,t}), \tag{31}$$

$$e^H_{i,k,h,\tau,t} = \mathrm{MLP}(r^H_{i,k,h,\tau,t}). \tag{32}$$

Here, $\delta^H_{i,k,h,\tau,t}$ denotes the temporal offset of the h-th historical window relative to the current window. Based on this embedding, the historical retrieval weight from the τ-th time step of the current proposal to the t-th time step of the h-th historical window is defined as

$$q^H_{i,k,\tau} = W^H_Q s_{i,k,\tau}, \tag{33}$$

$$k^H_{i,k,h,\tau,t} = W^H_K s_{i,h,t} + W^H_{Ke} e^H_{i,k,h,\tau,t}, \tag{34}$$

$$v^H_{i,k,h,\tau,t} = W^H_V s_{i,h,t} + W^H_{Ve} e^H_{i,k,h,\tau,t}, \tag{35}$$

$$\alpha^H_{i,k,h,\tau,t} = \mathrm{softmax}\left(\frac{q^H_{i,k,\tau}{}^T k^H_{i,k,h,\tau,t}}{\sqrt{d}}\right). \tag{36}$$

Accordingly, the historical local geometric context corresponding to the τ-th time step of the current proposal is obtained as

$$u^H_{i,k,\tau} = \sum_{h=1}^{H} \sum_{t=1}^{T_f} \alpha^H_{i,k,h,\tau,t} v^H_{i,k,h,\tau,t}. \tag{37}$$

Then, the complete local sequential feature of the current proposal is fused with the retrieved historical local geometric context, yielding the output of the local trajectory branch:

$$z^H_{i,k} = \mathrm{Fuse}(s_{i,k,\tau}, u^H_{i,k,\tau}). \tag{38}$$

This branch emphasizes the consistency of local path shape and motion direction between the current proposal and historical proposals within the overlapping intervals.

3.8. Training Objective

After obtaining the outputs of the high-level behavior consistency branch $z^I_{i,k}$ and the local trajectory branch $z^H_{i,k}$, the model first takes $z^I_{i,k}$ as the initial query of the second-stage backbone network and continues to update the mode representation over scene information

and interaction information, yielding the second-stage mode feature $z_{i,k}^{m,2}$. Then, the second-stage mode feature is fused with the local trajectory history context to obtain the final refined state:

$$o_{i,k} = \mathrm{MLP}(z_{i,k}^{m,2}, z_{i,k}^{H}). \tag{39}$$

Based on the aggregated state $o_{i,k}$, the model predicts the residual correction relative to the first-stage coarse proposal:

$$\Delta L_{i,k} = \mathrm{MLP}(o_{i,k}). \tag{40}$$

Accordingly, the final refined trajectory can be written as:

$$L_{i,k}^{2} = L_{i,k}^{1} + \Delta L_{i,k}. \tag{41}$$

Meanwhile, the mode probability distribution is also predicted from the same refined state:

$$\pi_{i}^{k} = \mathrm{softmax}(\mathrm{MLP}(o_{i,k})). \tag{42}$$

Finally, the loss function consists of the coarse proposal loss, the refined trajectory loss, and the classification loss, which are jointly optimized. The overall loss is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\mathrm{prop}} + \lambda_2 \mathcal{L}_{\mathrm{ref}} + \lambda_3 \mathcal{L}_{\mathrm{cls}}. \tag{43}$$

Here, $\mathcal{L}_{\mathrm{prop}}$ is used to constrain the geometric plausibility and mode coverage ability of the first-stage coarse proposals, $\mathcal{L}_{\mathrm{ref}}$ is used to directly optimize the geometric accuracy of the second-stage refined trajectories, and $\mathcal{L}_{\mathrm{cls}}$ is used to regularize the mode probability distribution so that the ground-truth corresponding mode is assigned a higher probability. The coefficients λ_1, λ_2, and λ_3 denote the weights of the three loss terms, respectively.

4. Experimentation and Evaluation

This section presents the experimental evaluation of the proposed PHR-Net. We first introduce the dataset, evaluation metrics, and implementation details. Then, we compare PHR-Net with representative trajectory prediction methods under both Top-1 and Top-6 evaluation settings. Finally, qualitative results, ablation studies, sensitivity analyses, operational efficiency, and temporal consistency verification are provided to comprehensively analyze the effectiveness and practicality of the proposed proposal-level historical retrieval framework.

4.1. Experimental Setup

4.1.1. Dataset

All experiments in this paper are conducted on the Argoverse 1 [29] motion forecasting dataset. This dataset is collected from real urban driving scenarios and provides high-definition (HD) maps as well as fine-grained trajectory annotations for traffic participants. Following the standard setting of this benchmark, trajectories are sampled at 10 Hz. In our experiments, the model observes the past 20 frames of the target agent and predicts its future motion over the next 30 frames. We focus on the single-target multimodal trajectory prediction task and thus output $K = 6$ candidate future trajectories for each target agent.

4.1.2. Evaluation Metrics

This study adopts the commonly used evaluation metrics from the Argoverse benchmark, including minADE, minFDE, and MR. Specifically, minADE measures the average displacement error of the predicted trajectory over the entire future horizon, reflecting the

overall trajectory fitting quality; minFDE considers only the displacement error at the final time step, placing greater emphasis on the accuracy of endpoint prediction; MR denotes the proportion of samples whose final displacement error exceeds 2.0 m, thereby characterizing the prediction failure rate. Under the multimodal prediction setting, the model outputs $K = 6$ candidate future trajectories for each target agent.

4.1.3. Implementation Details

The model is implemented based on PyTorch 2.1.0 and PyTorch Lightning 2.0.3 and trained on four NVIDIA RTX 4090 GPUs,detailed information is provided in Table 1. During training, the random seed is set to 1024, and the total number of training epochs is 32. AdamW is adopted as the optimizer, and the learning rate is scheduled with cosine annealing. For the scene interaction radius, both the agent-agent radius and the lane-agent radius are set to 50 m. In addition, we use 64 intention prototypes, top-8 prototype matching, and a historical window of length 3. Figure 2 visualizes the static intention prototype library used in the intention branch. The prototypes are obtained by applying K-means clustering to sliding-window trajectories in the training set under the agent-centric local frame. For data augmentation, horizontal flipping is applied during training with a probability of 0.5, and random masking with ratios of 0.05 and 0.2 is applied to the agent history tokens and map tokens, respectively.

Table 1. Experimental environment and implementation details.

Component	Specification
GPU	GeForce RTX 4090
CPU	AMD EPYC 7K62 48
Memory	64 GB DDR5
Storage	2 TB SSD
Operating system	Ubuntu 20.04
CUDA version	12.4
Deep learning framework	PyTorch 2.1.0

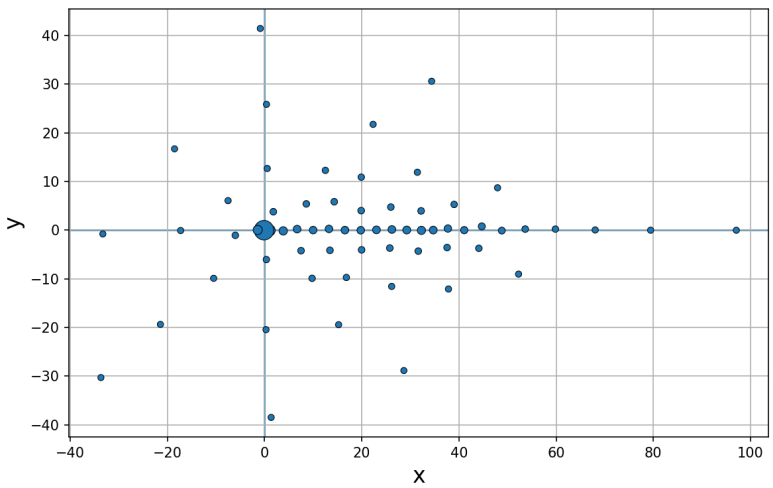

Figure 2. Distribution of static intention prototypes.The prototypes are obtained by clustering sliding-window trajectories in the training set under the agent-centric coordinate system. Each point represents a representative high-level motion pattern and provides a static behavioral prior for the intention branch.

4.2. Experimental Results

4.2.1. Quantitative Results

To verify the effectiveness of the proposed model, the following methods are selected as comparison baselines:

- LaneGCN [9]: A representative lane-graph modeling method. This method constructs a lane graph based on vectorized high-definition maps and models information propagation between lanes, between lanes and vehicles, and between vehicles through graph convolution.
- DenseTNT [30]: An anchor-free trajectory prediction method. This method estimates the probabilities of all dense candidate goal points on the road, then selects high-confidence goals through non-maximum suppression, and finally completes a full trajectory for each selected goal.
- Scene Transformer [23]: A scene-level Transformer prediction method. This method fuses map elements, historical trajectories, agent interactions, and temporal context information through attention mechanisms to achieve joint multi-agent trajectory prediction.
- HiVT [10]: A hierarchical vectorized Transformer method. This method divides the prediction process into two levels: local context extraction and global interaction modeling. It also introduces translation-invariant representation and rotation-invariant spatial learning modules to improve modeling efficiency and geometric robustness.
- Wayformer [25]: A trajectory prediction method based on a unified attention structure. This method adopts a homogeneous Transformer backbone to fuse historical trajectories, map semantics, and traffic participant interaction information, reducing handcrafted module design and improving multi-source scene information modeling capability.
- DCMS [18]: A training-enhancement method for continuous prediction stability. This method suppresses trajectory deviations under adjacent predictions or perturbed inputs through temporal consistency and spatial consistency constraints and uses multi-pseudo-target supervision to enhance multimodal learning capability.
- HPNet [19]: A modeling method that treats historical embedding representations as inputs for future timesteps. It introduces a historical prediction attention mechanism, using the agent interaction results from previous timesteps as reference information for the current prediction, thereby improving cross-timestep prediction consistency and stability.
- QCNet [16]: A trajectory prediction method based on the query-centric idea. This method reduces repeated computation in sliding-window online prediction through query-centric scene encoding and adopts a two-stage decoding structure that combines trajectory proposal generation and refinement to improve multimodal prediction accuracy and online inference efficiency.

The quantitative comparison results are reported in Tables 2 and 3. Table 2 presents the Top-1 evaluation results, where only the highest-confidence trajectory is used for evaluation. Table 3 reports the Top-6 evaluation results, where the metrics are computed using the best-matched trajectory among the top six candidate predictions. Therefore, the Top-1 setting mainly evaluates the accuracy of the most confident prediction, while the Top-6 setting reflects the ability of a method to cover multiple plausible future behaviors.

Table 2. Quantitative comparison under Top-1 evaluation on the Argoverse 1 benchmark.

Method	$\mathbf{minADE_1} \downarrow$	$\mathbf{minFDE_1} \downarrow$	$\mathbf{MR_1} \downarrow$
LaneGCN [9]	0.8703	1.3622	0.1620
DenseTNT [30]	0.8817	1.2815	0.1258
Scene Transformer [23]	0.8026	1.2321	0.1255
HiVT [10]	0.7735	1.1693	0.1267
HPNet [19]	0.7612	1.0986	0.1067
PHR-Net	0.7635	1.0834	0.1046

Table 3. Quantitative comparison under Top-6 evaluation on the Argoverse 1 benchmark.

Method	$\mathbf{minADE_6} \downarrow$	$\mathbf{minFDE_6} \downarrow$	$\mathbf{MR_6} \downarrow$
Wayformer [25]	0.7676	1.1616	0.1186
DCMS [18]	0.7659	1.1350	0.1094
QCNet [16]	0.7340	1.0666	0.1056
PHR-Net	0.7432	1.0804	0.1027

As shown in the two quantitative comparison tables, PHR-Net achieves competitive performance under both the Top-1 and Top-6 evaluation settings. Under the Top-1 setting, PHR-Net obtains the best minFDE_1 and MR_1 among the compared methods, indicating that the highest-confidence trajectory predicted by the model has better endpoint accuracy and a lower miss rate. Although its minADE_1 is slightly higher than that of HPNet, the improvements in minFDE_1 and MR_1 show that PHR-Net is more effective in reducing large endpoint deviations and severe prediction failures. Under the Top-6 setting, PHR-Net also achieves strong overall performance. Compared with Wayformer and DCMS, PHR-Net achieves lower errors on all three metrics, demonstrating the effectiveness of the proposed proposal-level refinement strategy. Compared with QCNet, PHR-Net obtains a lower MR_6, while its minADE_6 and minFDE_6 remain competitive. This indicates that the proposed method is particularly effective in reducing missed predictions, which is important for safety-critical trajectory forecasting tasks.

Compared with static-window prediction methods, such as LaneGCN [9], DenseTNT [30], Scene Transformer [23], HiVT [10], Wayformer [25], and QCNet [16], the advantage of PHR-Net is reflected not only in single-step prediction accuracy but also in its explicit use of historical information under continuous prediction scenarios. Although these methods have strong capabilities in scene representation, interaction modeling, and multimodal decoding, their outputs are still mainly determined independently by the input at the current time step. As a result, when deployed in a sliding-window manner, they are prone to frequent mode switching, discontinuities over overlapping temporal intervals, and local trajectory jitter. In contrast, PHR-Net introduces historical retrieval at the coarse proposal level, allowing the current prediction to depend not only on the current observation but also on previously generated candidate trajectories and their high-level mode information, thereby producing more stable continuous outputs.

Compared with DCMS [18] and HPNet [19], PHR-Net also shows certain advantages in standard prediction metrics. DCMS mainly improves prediction robustness through consistency constraints during training, while HPNet introduces historical prediction references at the embedding level. PHR-Net further performs historical modeling at the coarse proposal level, enabling the model to exploit trajectory shapes, endpoint destinations, and mode structures from historical proposals during the refinement stage. This proposal-level design provides more direct references for multimodal trajectory generation, thereby contributing to the improvements in minFDE and MR.

Overall, PHR-Net shows moderate improvements in minFDE and MR while maintaining reliable performance on minADE.

4.2.2. Qualitative Results

Figures 3 and 4 present qualitative results of continuous prediction under two representative types of scenarios: stable motion patterns and abrupt motion changes. The purpose of these visualizations is to examine whether PHR-Net can maintain cross-timestep consistency when the target agent follows a stable motion trend while still adapting smoothly when the future motion direction changes significantly.

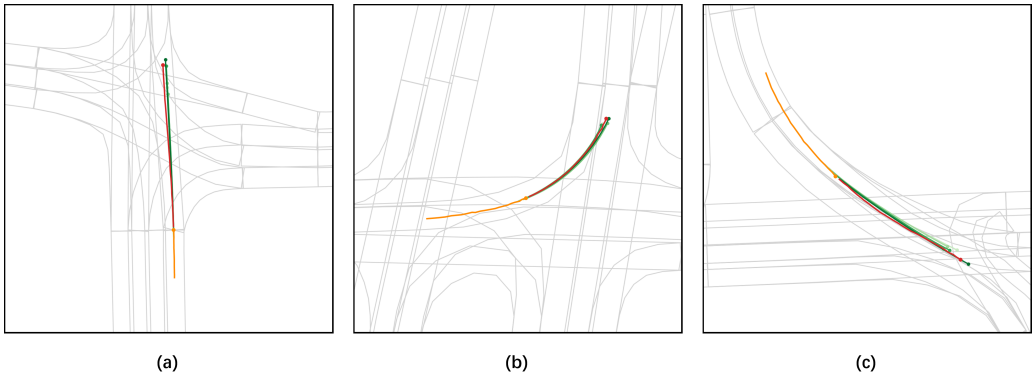

Figure 3. Qualitative results of continuous prediction in stable scenarios. (**a**) Straight-driving scenario, (**b**) turning scenario, and (**c**) curved-road scenario. The light gray lines denote the lane topology, the orange trajectory denotes the observed historical trajectory, the red trajectory denotes the ground-truth future trajectory, and the green trajectories denote the best predictions at adjacent consecutive time steps. The close alignment among green trajectories in the overlapping intervals indicates that PHR-Net can maintain temporal consistency under stable motion patterns.

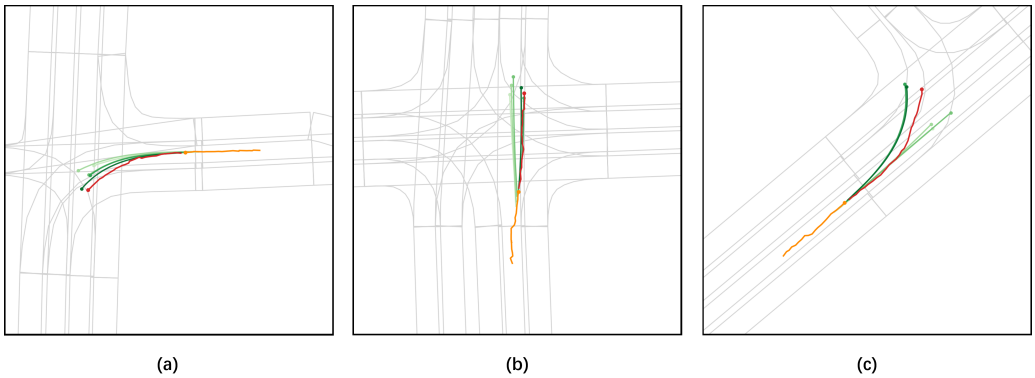

Figure 4. Qualitative results (**a**,**b**) show sudden lane-change scenarios, while (**c**) shows a sudden-turning scenario. The light gray lines denote the lane topology, the orange trajectory denotes the observed historical trajectory, the red trajectory denotes the ground-truth future trajectory, and the green trajectories denote the best predictions at adjacent consecutive time steps. Although the predictions are not forced to overlap completely, they evolve smoothly toward the new motion direction, PHR-Net progressively updates its prediction mode while preserving local continuity, indicating dynamic consistency under non-stationary driving behaviors.

Figure 3 shows the prediction results in stable scenarios over five consecutive prediction steps. Specifically, Figure 3a–c correspond to straight driving, turning, and curved-road driving scenarios, respectively. In these cases, the motion intention of the target agent remains relatively stable, and continuous prediction should therefore emphasize whether adjacent predictions remain coherent over their overlapping intervals on the real-time axis. As shown in the figure, the predictions at adjacent time steps are highly aligned

in the overlapping portions and follow an overall direction consistent with the ground truth (GT). This indicates that PHR-Net can effectively preserve cross-timestep consistency under stable motion patterns, thereby suppressing unnecessary mode switching and local trajectory jitter in continuous sliding-window prediction.

Figure 4 presents more challenging abrupt-change scenarios. Figure 4a,b show sudden lane-change cases, while Figure 4c shows a sudden-turning case. In these scenarios, the future motion of the target agent deviates from its previous motion trend, requiring the model to update its prediction mode in time while still maintaining local smoothness across consecutive predictions. It can be observed that the adjacent predictions no longer overlap as closely as in stable scenarios, but their evolution remains continuous and gradually moves toward the new motion direction indicated by the ground truth. This suggests that PHR-Net does not pursue forced consistency. Instead, it maintains dynamic consistency in non-stationary scenarios: when the environment and interaction patterns remain stable, adjacent predictions are encouraged to stay consistent; when the target behavior genuinely changes, the model can adapt its prediction mode while preserving local continuity. This is consistent with the notion of "dynamic correlation rather than forced alignment" discussed in HPNet [19].

4.3. Ablation Studies

To verify the effectiveness of each component in PHR-Net, we conduct ablation experiments on the Argoverse 1 validation set. Unless otherwise specified, all variants share the same Stage-1 coarse proposal generator, Stage-2 backbone network, training strategy, and hyperparameter settings. Only the historical modeling modules and their information flow paths are modified to ensure a fair comparison. If the intention branch is removed in a particular variant, a learnable mapping with the same dimensionality is used to encode the coarse proposals into the initial queries of Stage-2, thereby avoiding additional effects caused by differences in query form or parameter scale. The evaluation metrics follow those used in the main experiments, namely minADE, minFDE, and Miss Rate, to ensure comparability of the results.

4.3.1. Ablation Settings

We design five progressive ablation settings to systematically evaluate the way historical information is utilized, the necessity of the dual-branch design, and the contribution of the static prototype library.

Experiment A (base without history): Only the two-stage coarse-to-fine prediction framework is retained, without introducing any historical retrieval information. This setting serves as the overall performance baseline.

Experiment B (intention branch only): Based on the baseline model, the intention branch is added. High-level mode information is retrieved through static prototype matching and historical intention memory in order to verify the effect of high-level behavioral pattern modeling on the stability of continuous prediction.

Experiment C (Trajectory Branch only): Based on the baseline model, the Trajectory Branch is added. Local geometric context from historical proposals is introduced through sequence-level historical retrieval in order to validate the contribution of local trajectory continuity modeling.

Experiment D (intention + trajectory, without prototype library): Both the intention branch and the Trajectory Branch are enabled, while the static prototype library is removed and only historical intention memory retrieval is retained. This setting is used to analyze the impact of missing static behavioral priors under dual-branch collaboration.

Experiment E (PHR-Net, full model): The intention branch, Trajectory Branch, and static prototype library are all employed simultaneously, forming the complete model.

As shown in Table 4, after introducing historical information, all variants achieve performance improvements to varying degrees over the baseline without history, indicating that explicitly leveraging historical proposals is beneficial for improving prediction performance under the continuous prediction setting. Overall, the full PHR-Net achieves the best performance across all metrics, demonstrating the effectiveness of proposal-level historical retrieval in improving both prediction accuracy and stability. From the perspective of branch functionality, the variant with only the intention branch tends to show more noticeable improvement in the MR metric, suggesting that high-level mode retrieval helps reduce mode switching in continuous prediction. When the inputs at adjacent time steps change only slightly, historical intention information can provide a more stable behavioral prior for the current prediction, thereby reducing unnecessary mode transitions. In contrast, the variant with only the Trajectory Branch shows clearer advantages in minADE and minFDE, indicating that sequence-level historical retrieval can provide more direct local geometric references for the current prediction, thereby improving the continuity of trajectory shape and endpoint position. This suggests that, for overlapping intervals in continuous prediction, maintaining only high-level intention consistency is insufficient, and explicit modeling of local path details is also necessary. It is worth noting that when both the intention branch and the Trajectory Branch are enabled but the static prototype library is removed, the model can still combine high-level mode information with local geometric information, yet the overall improvement is typically less pronounced than that of the full model. This indicates that when relying solely on historical intention memory, the intention branch and the trajectory branch may still exhibit a certain degree of representational coupling, resulting in less distinct high-level mode information. After introducing the static prototype library, the intention branch acquires clearer behavioral pattern priors, thereby forming better functional complementarity with the Trajectory Branch: the former constrains the consistency of high-level behavioral modes, while the latter supplements the continuity of local path shapes and motion details.

Table 4. Ablation results on the Argoverse 1 validation set.

Model Configuration	Intent Branch	Trajectory Branch	Static Prototype Library	minADE ↓	minFDE ↓	MR ↓
Base				0.778	1.185	0.128
Base + Intent	✓		✓	0.733	1.024	0.084
Base + Traj		✓		0.675	0.976	0.106
Intent + Traj w/o Proto	✓	✓		0.670	0.937	0.094
PHR-Net	✓	✓	✓	0.641	0.858	0.067

4.3.2. Sensitivity Analysis

We further investigate the influence of two important hyperparameters in PHR-Net: the historical reference window size H and the number of static intention prototypes. These two factors control the amount of historical proposal information and the coverage of intention-level priors, respectively.

The length of the historical proposal memory determines the amount of cross-timestep reference information available for the current prediction. To evaluate its influence, we conduct a sensitivity analysis by varying the historical reference window size H. Figure 5a–c show the effects of H on minADE, minFDE, and MR, respectively. As H increases from 0 to 3, all three metrics generally decrease, indicating that historical proposal references are beneficial for trajectory geometry accuracy, endpoint prediction accuracy, and prediction reliability. It can be observed that minADE becomes nearly stable when H reaches 2 and

shows a slight increasing trend when H is further enlarged. Meanwhile, the improvements in minFDE and MR become marginal after H reaches 3. This suggests that the nearest few historical windows already provide most of the useful temporal reference information. Further increasing H may introduce redundant or outdated historical proposals and also increases the computational and memory overhead of the trajectory-history retrieval branch. Therefore, we set $H = 3$ in the final model, which provides a favorable trade-off among prediction accuracy, temporal consistency, and computational efficiency.

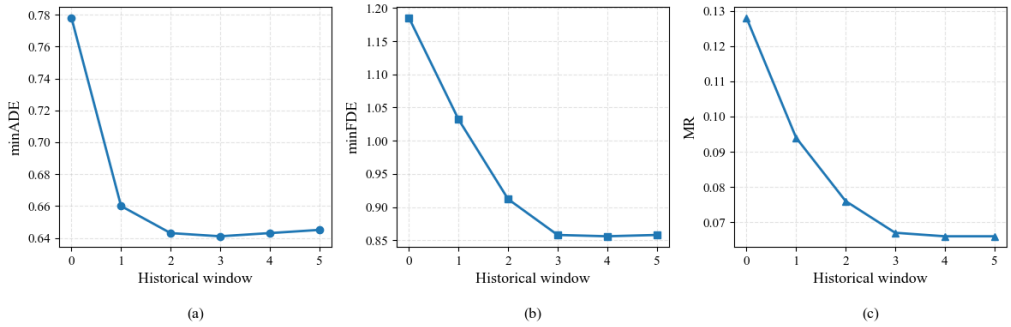

Figure 5. Sensitivity analysis of the historical reference window size. (**a**) minADE, (**b**) minFDE, and (**c**) MR under different historical window sizes. A window size of 0 denotes the model without historical proposal retrieval.

In addition to the historical reference window size, the scale of the static intention prototype library also affects the coverage of high-level behavioral patterns. To determine an appropriate number of static intention points, we vary the prototype library size and analyze its influence on prediction performance. Figure 6a–c show the influence of the number of static intention points on minADE, minFDE, and MR, respectively. As the number of static intention points increases, all three metrics gradually decrease, indicating that static intention priors contribute positively to both trajectory geometry modeling and high-level intention modeling. When the number of intention points is small, the prototype library cannot sufficiently cover all potential motion intentions of the target agent, leading to relatively weaker performance. When the number of intention points exceeds 64, the improvements in all metrics become marginal, suggesting that the common motion patterns in the training set have already been well covered. Therefore, the final model adopts 64 static intention points.

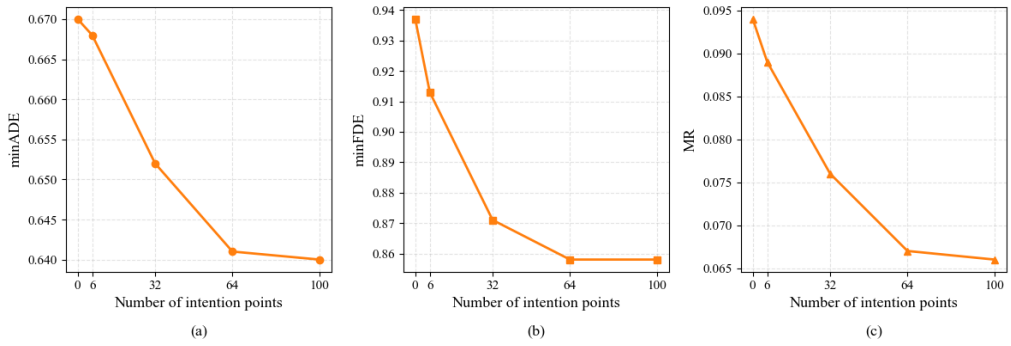

Figure 6. Sensitivity analysis of the number of static intention points. (**a**) minADE, (**b**) minFDE, and (**c**) MR under different numbers of intention points. The setting with 0 intention points denotes the model without the static prototype library.

4.3.3. Operational Efficiency Analysis

To evaluate the additional computational overhead introduced by the historical retrieval module, we compare PHR-Net and HPNet in terms of the number of parameters,

inference latency, FPS, and peak GPU memory under the same inference setting. All evaluations are conducted on the Argoverse 1 test set with batch size 1 to simulate online inference, using a single NVIDIA GeForce RTX 4090 GPU.

As shown in Table 5, PHR-Net has fewer trainable parameters than HPNet, with an 11.52% reduction in parameter count. This indicates that the proposed historical retrieval design does not increase the model scale. However, since PHR-Net introduces proposal-level historical retrieval during inference, especially in the Trajectory Branch where current proposal sequences retrieve local geometric references from historical proposal sequences, it brings a moderate increase in runtime cost. Specifically, the average inference latency increases by 15.61%, from 41.841 ms to 48.372 ms, and the corresponding FPS decreases from 23.900 to 20.673. Meanwhile, the peak GPU memory increases by 9.15%, from 1.770 GB to 1.932 GB. Overall, the additional overhead remains limited under the online inference setting, indicating that PHR-Net improves temporal consistency in continuous prediction with acceptable computational cost.

Table 5. Operational efficiency comparison under the same Argoverse 1 online inference setting.

Method	Params (M) ↓	Latency (ms) ↓	FPS ↑	Peak Memory (GB) ↓
HPNet	4.070	41.841	23.900	1.770
PHR-Net	3.601	48.372	20.673	1.932

4.3.4. Consistency Verification on the Overlapping Portions of Adjacent Predictions

To further evaluate the temporal consistency of PHR-Net in continuous prediction scenarios, we adopt summed ADE to quantify the error of adjacent predictions within their overlapping interval on the real-time axis. Given a continuous prediction scenario, let the current time step be t_0. Based on the observation window $[t_0 - T_{\text{obs}}, t_0]$, the model outputs a future trajectory $\hat{Y}_{t_0}$. At the next time step, $t_1 = t_0 + \Delta t$, the model takes the observation window $[t_1 - T_{\text{obs}}, t_1]$ as input and outputs another future trajectory $\hat{Y}_{t_1}$. These two predictions share an overlapping interval of length $T_{\text{fut}} - \Delta t$ on the real future timeline. For each sample, we extract the trajectory points from two adjacent predictions over the overlapping time steps, first perform Hungarian matching across modes, and then compute the average Euclidean distance between the matched trajectories.

As shown in Table 6, our method achieves a summed ADE of 2.08, which is substantially lower than that of the model without historical reference (3.00) and also lower than that of HPNet [19] (2.25). This result indicates that, compared with introducing historical information only at the feature level, proposal-level historical retrieval can more effectively exploit the mode structure and geometric information contained in historical predictions, thereby producing more consistent outputs within the overlapping interval after alignment on the real-time axis. This further validates the design motivation of the proposed method for continuous prediction scenarios, namely that both high-level behavioral mode stability and local trajectory continuity should be explicitly modeled at the proposal level.

Table 6. Comparison of summed ADE in overlapping intervals.

Method	Summed ADE ↓
Baseline w/o historical reference	3.00
HPNet [19]	2.25
PHR-Net	2.08

5. Conclusions

This paper addresses the problem of non-stationary temporal consistency in continuous trajectory prediction for autonomous driving and proposes PHR-Net, a proposal-level historical retrieval framework. PHR-Net treats the Stage-1 coarse proposals as historical memory and performs consistency-aware refinement of current predictions at the proposal level. By decomposing historical consistency into two complementary aspects, namely high-level intention stability and local trajectory geometric continuity, the proposed intention branch and trajectory branch can jointly reduce unnecessary mode switching and improve the smoothness of adjacent predictions.

Experimental results on the Argoverse 1 benchmark show that PHR-Net achieves competitive prediction performance under both Top-1 and Top-6 evaluation settings. Under the Top-1 setting, compared with the strongest baseline, PHR-Net reduces minFDE and MR by 1.38% and 1.97%, respectively, indicating moderate improvements in endpoint accuracy and prediction reliability. Under the Top-6 setting, PHR-Net reduces MR by 2.75% compared with the strongest competing method. More importantly, in the overlapping-interval consistency evaluation, PHR-Net reduces the summed ADE by 30.67% compared with the variant without historical reference and by 7.56% compared with HPNet. These quantitative results show that proposal-level historical retrieval can improve not only standard prediction reliability but also cross-timestep temporal consistency in continuous prediction.

From a practical perspective, temporally consistent prediction results can help online autonomous driving systems reduce prediction jitter and unstable mode switching across consecutive sliding windows, thereby providing more reliable inputs for downstream planning, risk assessment, braking, yielding, and decision-making modules. Therefore, PHR-Net provides a structured and interpretable solution for improving the stability and reliability of continuous trajectory prediction.

Although PHR-Net improves continuous prediction stability, several limitations remain. First, the historical memory length and update strategy affect retrieval quality: a short memory may miss useful historical references, whereas an overly long memory may introduce outdated information and additional computational overhead. Second, sequence-level historical retrieval brings extra GPU memory and latency costs during online inference. Future work will further explore adaptive memory update strategies, sparse memory mechanisms, hierarchical retrieval, and approximate attention methods to improve the efficiency and robustness of proposal-level historical retrieval in large-scale online deployment.

Author Contributions: Conceptualization, B.Z. and M.X.; Methodology, B.Z. and M.X.; Software, B.Z.; Validation, B.Z.; Formal analysis, B.Z.; Investigation, B.Z.; Writing—original draft, B.Z.; Writing—review and editing, M.X.; Visualization, B.Z.; Supervision, M.X.; Project administration, M.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original data presented in the study are openly available in [29].

Acknowledgments: The authors would like to thank the editors and the reviewers for their valuable time and constructive comments.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MLP	Multi-Layer Perceptron
CNN	Convolutional Neural Network

GNN	Graph Neural Network
GRU	Gated Recurrent Unit
HD	High-Definition
GT	Ground Truth
AdamW	Adaptive Moment Estimation with decoupled Weight Decay
minADE	Minimum Average Displacement Error
minFDE	Minimum Final Displacement Error
MR	Miss Rate
summed ADE	Summed Average Displacement Error
FPS	Frames Per Second
Top-1	Evaluation using only the highest-confidence predicted trajectory
Top-6	Evaluation using the best-matched trajectory among the top six predicted trajectories

Appendix A. Detailed Flowchart

To further clarify the complete computation process of PHR-Net, we provide a detailed algorithm flowchart in Figure A1. Since the full mathematical notation contains repeated agent, mode, and time indices, the flowchart adopts compact notation to improve readability. The correspondence between the compact notation in the flowchart and the full notation used in the main text is summarized in Table A1.

Table A1. Correspondence between compact notation in Figure A1 and full notation used in the main text.

Compact Notation in Flowchart	Full Notation or Meaning in the Main Text
X_i	Historical trajectory of target agent i
N_i	Neighboring-agent set of target agent i
M_i	Local map elements of target agent i
z^S	Scene-attention feature $z^s_{i,k}$
z^A	Agent-attention feature $z^a_{i,k}$
z^M	Mode-attention feature $z^m_{i,k}$
L^1	Stage-1 coarse proposal $L^1_{i,k}$
B	Historical proposal memory of target agent i
g	High-level behavior representation $g_{i,k}$
β^I	Prototype matching weight $\beta^I_{i,k,r}$
z^I	Retrieved intention-history feature $z^I_{i,k}$
$z^{M,2}$	Second-stage mode feature $z^{m,2}_{i,k}$
s	Proposal sequence encoding $s_{i,k,\tau}$
u^H	Retrieved trajectory-history context $u^H_{i,k,\tau}$
z^H	Trajectory-history feature $z^H_{i,k}$
o	Fused feature $o_{i,k}$
ΔL	Residual correction $\Delta L_{i,k}$
L^2	Final refined trajectory $L^2_{i,k}$
π	Mode probability π^k_i
$\mathcal{L}$	Total training objective

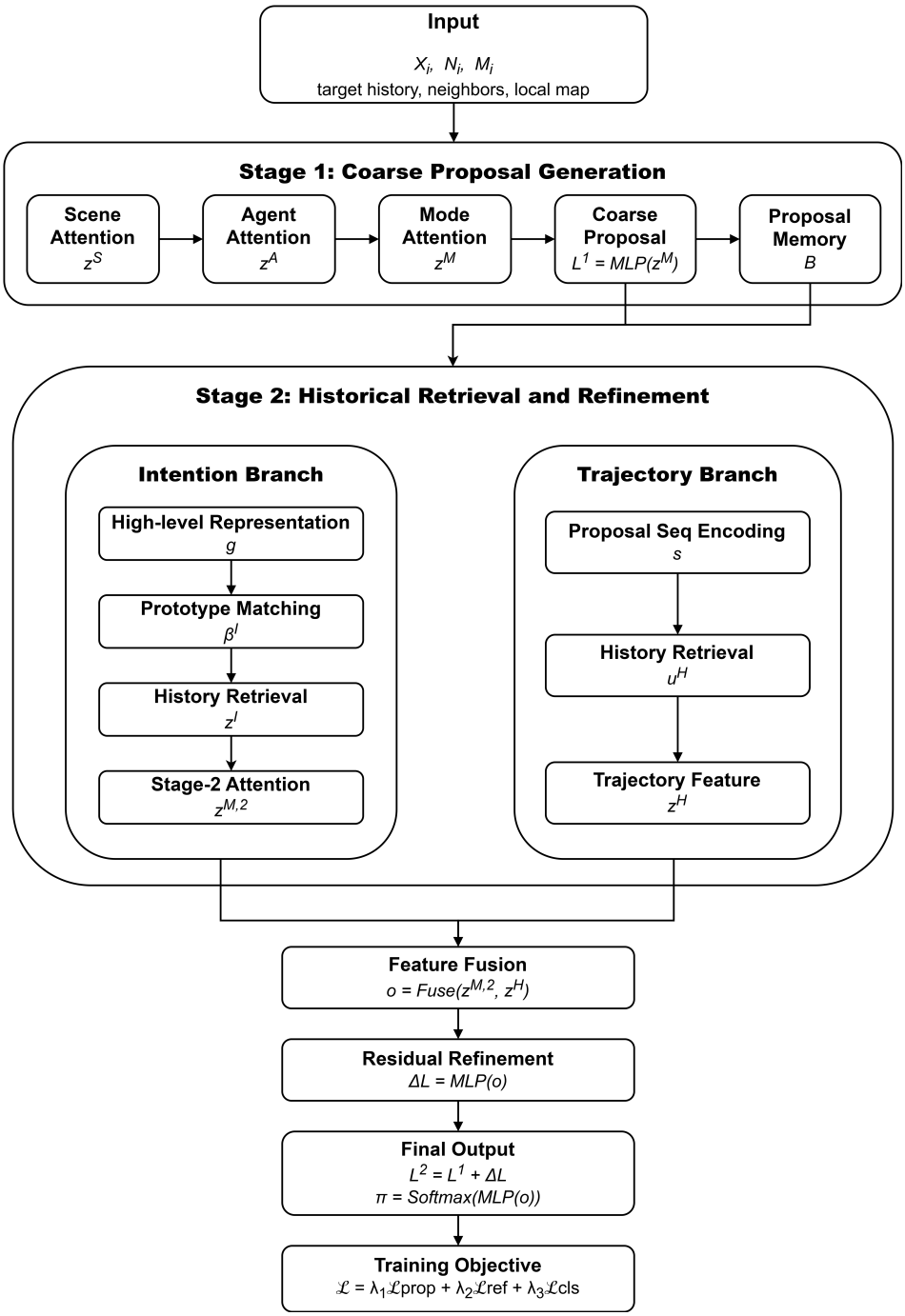

Figure A1. Detailed flowchart of PHR-Net. The figure illustrates the complete two-stage computation process of PHR-Net. In Stage 1, scene attention, agent attention, and mode attention are sequentially applied to generate coarse proposals, which are then stored in the historical proposal memory. In Stage 2, the current coarse proposals are refined through two complementary branches. For readability, compact variable notation is used in the flowchart, and the correspondence between compact notation and full notation is summarized in Table A1.

References

1. Kalman, R.E. A New Approach to Linear Filtering and Prediction Problems. *J. Basic Eng.* **1960**, *82*, 35–45. [CrossRef]
2. Helbing, D.; Molnár, P. Social force model for pedestrian dynamics. *Phys. Rev. E* **1995**, *51*, 4282–4286. [CrossRef] [PubMed]
3. Alahi, A.; Goel, K.; Ramanathan, V.; Robicquet, A.; Fei-Fei, L.; Savarese, S. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 961–971. [CrossRef]

4. Lee, N.; Choi, W.; Vernaza, P.; Choy, C.B.; Torr, P.H.S.; Chandraker, M. DESIRE: Distant Future Prediction in Dynamic Scenes with Interacting Agents. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 2165–2174. [CrossRef]
5. Gupta, A.; Johnson, J.; Fei-Fei, L.; Savarese, S.; Alahi, A. Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 2255–2264. [CrossRef]
6. Ivanovic, B.; Pavone, M. The Trajectron: Probabilistic Multi-Agent Trajectory Modeling with Dynamic Spatiotemporal Graphs. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, South Korea, 27 October–2 November 2019; pp. 2375–2384. [CrossRef]
7. Salzmann, T.; Ivanovic, B.; Chakravarty, P.; Pavone, M. Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data. In *Computer Vision — ECCV 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer International Publishing: Cham, Switzerland , 2020; pp. 683–700. [CrossRef]
8. Gao, J.; Sun, C.; Zhao, H.; Shen, Y.; Anguelov, D.; Li, C.; Schmid, C. VectorNet: Encoding HD Maps and Agent Dynamics From Vectorized Representation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 11522–11530. [CrossRef]
9. Liang, M.; Yang, B.; Hu, R.; Chen, Y.; Liao, R.; Feng, S.; Urtasun, R. Learning Lane Graph Representations for Motion Forecasting. In *Computer Vision — ECCV 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 541–556. [CrossRef]
10. Zhou, Z.; Ye, L.; Wang, J.; Wu, K.; Lu, K. HiVT: Hierarchical Vector Transformer for Multi-Agent Motion Prediction. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022, pp. 8813–8823. [CrossRef]
11. Jia, X.; Wu, P.; Chen, L.; Liu, Y.; Li, H.; Yan, J. HDGT: Heterogeneous Driving Graph Transformer for Multi-Agent Trajectory Prediction via Scene Encoding. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 13860–13875. [CrossRef] [PubMed]
12. Chai, Y.; Sapp, B.; Bansal, M.; Anguelov, D. MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction. In *Proceedings of the Conference on Robot Learning*; Kaelbling, L.P., Kragic, D., Sugiura, K., Eds.; (Proceedings of Machine Learning Research); PMLR: Online, 2020; Volume 100, pp. 86–99. Available online: https://proceedings.mlr.press/v100/chai20a.html (accessed on 10 May 2026).
13. Phan-Minh, T.; Grigore, E.C.; Boulton, F.A.; Beijbom, O.; Wolff, E.M. CoverNet: Multimodal Behavior Prediction Using Trajectory Sets. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 14062–14071. [CrossRef]
14. 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 *Proceedings of the 2020 Conference on Robot Learning*; Kober, J., Ramos, F., Tomlin, C., Eds.; (Proceedings of Machine Learning Research) ; PMLR: Online, 2021; Volume 155, pp. 895–904. Available online: https://proceedings.mlr.press/v155/zhao21b.html (accessed on 10 May 2026).
15. Shi, S.; Jiang, L.; Dai, D.; Schiele, B. Motion Transformer with Global Intention Localization and Local Movement Refinement. In *Proceedings of the Advances in Neural Information Processing Systems*; Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., Oh, A., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2022; pp. 6531–6543. Available online: https://proceedings.neurips.cc/paper_files/paper/2022/file/2ab47c960bfee4f86dfc362f26ad066a-Paper-Conference.pdf (accessed on 10 May 2026).
16. Zhou, Z.; Wang, J.; Li, Y.H.; Huang, Y.K. Query-Centric Trajectory Prediction. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 17863–17873. [CrossRef]
17. Jiang, C.M.; Cornman, A.; Park, C.; Sapp, B.; Zhou, Y.; Anguelov, D. MotionDiffuser: Controllable Multi-Agent Motion Prediction Using Diffusion. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 9644–9653. [CrossRef]
18. Ye, M.; Xu, J.; Xu, X.; Cao, T.; Chen, Q. DCMS: Motion Forecasting with Dual Consistency and Multi-Pseudo-Target Supervision, *arXiv* **2022**, arXiv:2204.05859 .
19. Tang, X.; Kan, M.; Shan, S.; Ji, Z.; Bai, J.; Chen, X. HPNet: Dynamic Trajectory Forecasting with Historical Prediction Attention. In Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 17–21 June 2024; pp. 15261–15270. [CrossRef]
20. Zang, Z.; Zhang, X.; Gong, X.; Song, J.; Yu, R.; Gong, J. A spatio-temporal trajectory planning framework for AGVs based on motion primitive and dynamic programming in off-road environments. *Adv. Eng. Inform.* **2026**, *69*, 103934. [CrossRef]
21. Casas, S.; Luo, W.; Urtasun, R. IntentNet: Learning to Predict Intention from Raw Sensor Data. In *Proceedings of the 2nd Conference on Robot Learning*; Billard, A.; Dragan, A.; Peters, J.; Morimoto, J., Eds.; PMLR: Online, 2018; Volume 87, pp. 947–956. Available online: https://proceedings.mlr.press/v87/casas18a.html (accessed on 10 May 2026).

22. Hong, J.; Sapp, B.; Philbin, J. Rules of the Road: Predicting Driving Behavior with a Convolutional Model of Semantic Interactions. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 8446–8454. [CrossRef]
23. Ngiam, J.; Vasudevan, V.; Caine, B.; Zhang, Z.; Chiang, H.T.L.; Ling, J.; Roelofs, R.; Bewley, A.; Liu, C.; Venugopal, A.; et al. Scene Transformer: A unified architecture for predicting future trajectories of multiple agents. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022. Available online: https://openreview.net/forum?id=Wm3EA5OlHs (accessed on 10 May 2026).
24. Yuan, Y.; Weng, X.; Ou, Y.; Kitani, K. AgentFormer: Agent-Aware Transformers for Socio-Temporal Multi-Agent Forecasting. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 11–17 October 2021; pp. 9793–9803. [CrossRef]
25. Nayakanti, N.; Al-Rfou, R.; Zhou, A.; Goel, K.; Refaat, K.S.; Sapp, B. Wayformer: Motion forecasting via simple and efficient attention networks. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 2980–2987. [CrossRef]
26. Sun, Q.; Huang, X.; Gu, J.; Williams, B.C.; Zhao, H. M2I: From Factored Marginal Trajectory Prediction to Interactive Prediction. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 6533–6542. [CrossRef]
27. Marchetti, F.; Becattini, F.; Seidenari, L.; Del Bimbo, A. MANTRA: Memory Augmented Networks for Multiple Trajectory Prediction. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 7141–7150. [CrossRef]
28. Rhinehart, N.; Kitani, K.M.; Vernaza, P. r2p2: A ReparameteRized Pushforward Policy for Diverse, Precise Generative Path Forecasting. In *Computer Vision – ECCV 2018*; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 794–811. [CrossRef]
29. Chang, M.F.; Lambert, J.; Sangkloy, P.; Singh, J.; Bak, S.; Hartnett, A.; Wang, D.; Carr, P.; Lucey, S.; Ramanan, D.; et al. Argoverse: 3D Tracking and Forecasting with Rich Maps. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 8740–8749. [CrossRef]
30. Gu, J.; Sun, C.; Zhao, H. DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 11–17 October 2021; pp. 15283–15292. [CrossRef]