sensors

MDPI

Article

Multimodal Trajectory Prediction for Diverse Vehicle Types in Autonomous Driving with Heterogeneous Data and Physical Constraints

Maoning Ge [1,*], Kento Ohtani [1], Ming Ding [2], Yingjie Niu [1], Yuxiao Zhang [3] and Kazuya Takeda [1,4]

[1] Graduate School of Informatics, Nagoya University, Furo-cho, Chikusa-Ward, Nagoya 464-8601, Japan
[2] Zhejiang Fubang Technology Inc., Ningbo R&D Campus Block A, Ningbo 315048, China
[3] RoboSense Technology Co., Ltd., 701 Block B, 800 Naxian Road, Pudong, Shanghai 200131, China
[4] Tier IV Inc., Nagoya University Open Innovation Center, 1-3, Mei-eki 1-chome, Nakamura-Ward, Nagoya 450-6610, Japan
* Correspondence: maoning.ge@g.sp.m.is.nagoya-u.ac.jp

Abstract: The accurate prediction of vehicle behavior is crucial for autonomous driving systems, impacting their safety and efficiency in complex urban environments. To address the challenge of multi-agent trajectory prediction, we propose a novel model integrating multiple input modalities, including historical trajectories, map data, vehicle features, and interaction information. Our approach employs a Conditional Variational Autoencoder (CVAE) framework with a decoder that predicts control actions using the Gaussian Mixture Model (GMM) and then converts these actions into dynamically feasible trajectories through a bicycle model. Evaluated on the nuScenes dataset, the model achieves great performance across key metrics, including minADE_5 of 1.26 and minFDE_5 of 2.85, demonstrating robust performance across various vehicle types and prediction horizons. These results indicate that integrating multiple data sources, physical models, and probabilistic methods significantly improves trajectory prediction accuracy and reliability for autonomous driving. Our approach generates diverse yet realistic predictions, capturing the multimodal nature of future outcomes while adhering to Physical Constraints and vehicle dynamics.

Keywords: multi-agent trajectory prediction; multimodal learning; Conditional Variational Autoencoder; Gaussian Mixture Model; autonomous driving

Citation: Ge, M.; Ohtani, K.; Ding, M.; Niu, Y.; Zhang, Y.; Takeda, K. Multimodal Trajectory Prediction for Diverse Vehicle Types in Autonomous Driving with Heterogeneous Data and Physical Constraints. *Sensors* **2024**, *24*, 7323. https://doi.org/10.3390/s24227323

Academic Editors: Araceli Sanchis de Miguel and Agapito Ledezma Espino

Received: 23 October 2024
Revised: 13 November 2024
Accepted: 14 November 2024
Published: 16 November 2024

1. Introduction

The accurate prediction of vehicle behavior is crucial for autonomous driving systems, as it directly impacts their safety and efficiency [1–3]. By anticipating the future actions of surrounding road users, self-driving cars can effectively avoid collisions and navigate traffic smoothly. This capability is essential to establishing trust and integrating autonomous vehicles on public roads [4].

Humans exhibit an innate capacity for social cognition, or "theory of mind", which allows them to comprehend and predict the intentions of others [5,6]. This enables them to anticipate the actions of those around them, facilitating smooth and successful navigation. Autonomous vehicles can draw inspiration from the way humans navigate complex social environments [7]. For autonomous systems to operate safely in real-world environments, they must demonstrate a level of social awareness and prediction similar to that of humans [8,9].

As illustrated in Figure 1, accurately predicting the behaviors of surrounding traffic, such as during overtaking maneuvers, plays a critical role in guiding the planning and decision-making of autonomous vehicles. By anticipating the actions of nearby vehicles like sedans and trucks, the system can safely and efficiently execute lane changes and other maneuvers. The differences in merging behavior between sedans and trucks are mainly

due to their braking capability and size. Trucks have longer braking distances and larger physical dimensions, making it more challenging for them to slow down quickly and allow space for merging vehicles. In contrast, sedans are smaller and more agile, typically making it easier for the autonomous vehicle to merge safely and quickly. Given these differences, the ability to accurately predict multi-agent behavior has become critical for developing safe and efficient autonomous driving systems [10].

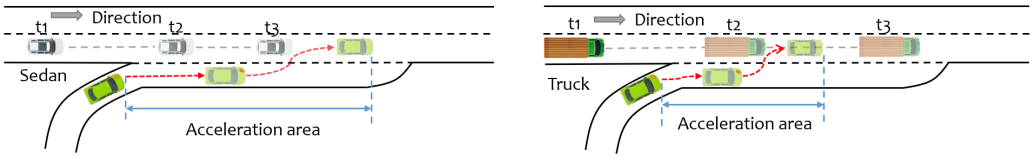

Figure 1. Trajectory prediction for lane merging maneuvers when merging in front of a sedan (**left**) and a truck (**right**), illustrating the predicted paths, acceleration areas, and decision-making process of the autonomous vehicle.

Current methods for multi-agent behavior prediction range from traditional kinematic models to advanced learning-based approaches [11,12]. Traditional methods, such as physics-based models, predict future positions using basic motion parameters such as velocity and acceleration [13–15]. Although straightforward, these models often fall short in complex and dynamic environments because they assume that agents move independently and neglect interaction dynamics. Rule-based models attempt to incorporate interaction heuristics, but their rigidity limits their adaptability to real-world scenarios [16,17].

Recent years have seen a shift toward learning-based models, which leverage data-driven techniques to capture complex patterns and interactions [11,12,18,19]. These approaches show promise in handling the intricacies of multi-agent scenarios. However, they still face significant challenges. Many current methods struggle to fully address real-world complexities, often overlooking crucial factors such as dynamic constraints, the ego agent's motion, and rich environmental data from sources like high-definition maps and lidar sensors. Addressing these limitations is crucial for developing robust and reliable prediction systems capable of enhancing the safety and efficiency of autonomous driving in diverse real-world scenarios.

To better handle the uncertainty in vehicle behavior, many studies have explored probabilistic approaches for behavior prediction. For instance, Dynamic Bayesian Networks (DBNs) have been successfully applied in highway driving scenarios to predict vehicle maneuvers, demonstrating robustness and adaptability through empirical validation [20]. These DBN methods, which incorporate driver uncertainty and vehicle dynamics, have proven effective across various driving conditions [21]. Additionally, probabilistic architectures for long-term vehicle trajectory prediction are designed to manage uncertainties and provide flexible predictions in complex, multimodal traffic environments [22]. Recently, models such as the Conditional Variational Autoencoder (CVAE) [23–27] and Gaussian Mixture Model (GMM) [26,28] have shown potential in generating diverse and realistic trajectories, with CVAE producing varied possible outcomes from similar conditions and GMM capturing a range of trajectory patterns as a mixture of distributions. These probabilistic models underscore the importance of handling the inherent uncertainties and variability in multi-agent scenarios, offering a promising direction for improving prediction accuracy and safety in autonomous driving systems.

To address the complexity and uncertainty in multi-agent behavior prediction, this paper proposes a multimodal trajectory prediction model based on CVAE. The main contributions of this work are as follows:

1. The integration of heterogeneous data sources to improve prediction accuracy and diversity, generating trajectories for different vehicle types. The proposed model leverages various heterogeneous data sources, including high-definition maps, vehicle features, and interaction data between road agents, to generate customized trajectory

predictions. By incorporating these contextual cues, the model captures the complex motion dynamics and interaction patterns of different road agents.
2. The proposed model uses a CVAE structure, with control variables as the predicted output, and represents the predicted trajectories using a GMM. This approach captures multiple plausible future trajectories and quantifies the uncertainty in the predictions, improving the model's ability to handle complex traffic environments.
3. A bicycle model is incorporated as a Physical Constraint. The model introduces a bicycle model as a Physical Constraint in the learning-based framework for multi-agent trajectory prediction, ensuring the generated trajectories are physically feasible and realistic.

2. Related Works

2.1. Traditional Trajectory Prediction Methods

Traditional approaches to trajectory prediction have relied on simplified models and predefined rules, often failing to capture the complexity of real-world driving scenarios. These methods can be broadly categorized into two main types: physics-based models and rule-based approaches.

Physics-based models, including kinematic and dynamic models, predict future trajectories based on fundamental principles and observed motion parameters. They extrapolate future positions by considering current velocity, acceleration, and heading, assuming relatively constant motion patterns [13–15]. Although straightforward to implement, these models struggle to account for the unpredictable nature of human drivers and the influence of external factors such as road geometry and interactions with other vehicles.

Rule-based approaches utilize predefined heuristics to anticipate agent behavior [16,17]. These rules are often derived from traffic laws, common driving practices, or expert knowledge. However, the rigidity of these rules limits their ability to adapt to diverse and dynamic situations, especially when encountering unexpected maneuvers or complex interactions.

The inherent limitations of traditional methods stem from their inability to adequately model the complex factors influencing real-world driving behavior. They often neglect interaction dynamics, struggle with dynamic environments, and lack adaptability. These shortcomings underscore the need for more sophisticated models capable of learning from data and adapting to the complexities of real-world driving.

2.2. Learning-Based Trajectory Prediction Methods

In response to the limitations of traditional methods, learning-based approaches have gained significant traction in recent years. These methods leverage the power of machine learning to extract patterns and relationships from data, enabling them to handle the complexities of multi-agent trajectory prediction more effectively. Some prominent learning-based methods include Recurrent Neural Networks (RNNs), Generative Adversarial Networks (GANs), Conditional Variational Autoencoders (CVAEs), and Graph Neural Networks (GNNs).

RNNs excel at modeling sequential data, which makes them well suited for trajectory prediction [29–36]. By incorporating feedback loops, they can capture temporal dependencies in agent movements, learning how past positions influence future trajectories. This allows them to predict future states based on the historical movement patterns of agents.

GANs consist of two neural networks—a generator and a discriminator—trained in an adversarial manner [9,37–39]. The generator learns to produce realistic trajectories, while the discriminator aims to distinguish between real and generated trajectories. This adversarial training process pushes the generator to produce increasingly realistic and diverse trajectory predictions.

CVAEs are a type of variational autoencoder that conditions the encoding and decoding process on additional information, such as context or agent state. In trajectory prediction, they can model the inherent uncertainty and multi-modality of future trajecto-

ries by generating multiple possible outcomes given the same input conditions [23–27]. This probabilistic approach is especially useful in uncertain and dynamic driving environments.

GNNs are specifically designed to operate on graph-structured data, which makes them ideal for capturing interactions between agents [33,40–42]. By representing agents and their relationships as nodes and edges in a graph, they can learn complex interaction patterns and predict future trajectories based on the influence of neighboring agents.

The strengths of learning-based models include their ability to capture complex agent interactions and adapt to dynamic scenarios, improving accuracy in complex, real-world settings. However, these models face challenges such as computational complexity in training and deployment and the need for large, diverse datasets. They may also struggle to generalize to unfamiliar environments or scenarios different from their training data and often fall short of integrating real-world Physical Constraints or combining diverse data sources.

2.3. Incorporating Physical Constraints and Dynamics

While learning-based approaches have shown promise in multi-agent trajectory prediction, accurately modeling the physical world remains a significant challenge. Integrating Physical Constraints and dynamics, such as vehicle dynamics and road geometry, is crucial for developing realistic and reliable prediction models [43].

Recent research has explored the incorporation of vehicle dynamics into trajectory prediction frameworks [44,45]. These models consider factors like vehicle dimensions, steering angles, and tire slip to provide a more realistic representation of vehicle motion. By constraining predicted trajectories to physically plausible paths, these approaches aim to improve prediction accuracy and safety.

Studies have also highlighted the importance of considering vehicle-specific attributes, such as vehicle type (e.g., car, truck, motorcycle) and size, in prediction models [18]. Different vehicle types exhibit distinct motion characteristics and constraints, influencing their trajectory decisions. Incorporating this information can lead to more accurate and context-aware predictions [46,47].

3. Methodology

Our proposed model, MTP-HPC (Multimodal Trajectory Prediction with Heterogeneous Data and Physical Constraints), addresses the challenges of multi-agent trajectory prediction. The model integrates multiple data sources and advanced machine learning techniques to improve prediction accuracy for diverse vehicle types. The architecture of MTP-HPC consists of two main components: a feature extraction module and a trajectory generation module.

In the feature extraction stage, the model processes agent historical trajectories using Long Short-Term Memory (LSTM) to capture temporal dynamics. Simultaneously, vehicle features are processed through an embedding layer followed by a Multi-Layer Perceptron (MLP) to extract relevant features. To capture environmental context, Convolutional Neural Networks (CNNs) are employed to extract features from high-definition map data, which are then refined using fully connected (FC) layers. An attention-based interaction network models the interactions between various agents, such as vehicles and pedestrians. After feature extraction, the different data sources are concatenated into a unified 128-dimensional representation, forming the input for trajectory prediction.

In the trajectory generation stage, a CVAE framework is used to address the inherent uncertainty and multi-modal nature of future trajectories. The latent variable z is processed through an FC layer and then passed to the decoder. The decoder integrates a Gated Recurrent Unit (GRU), a GMM, and a kinematic bicycle model to produce realistic and diverse trajectory predictions. The GMM predicts control signals such as acceleration a_k^t and steering angle δ_k^t, which are used by the kinematic bicycle model to generate physically feasible trajectories. This approach ensures that the generated trajectories are diverse while adhering to environmental constraints and incorporating prediction uncertainty.

Figure 2 illustrates the complete architecture of our model, where each module collaborates to effectively capture the complex behavior of agents in multi-agent traffic scenarios.

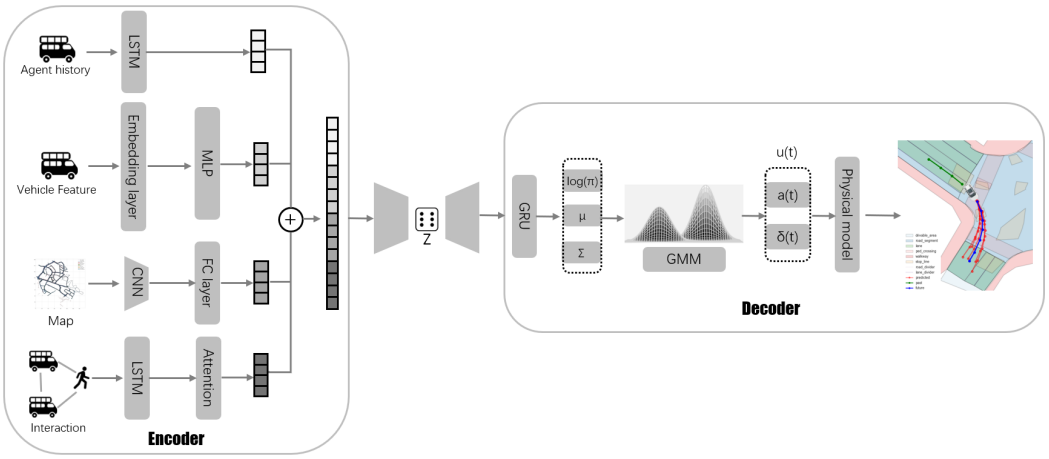

Figure 2. Architecture of the proposed MTP-HPC model, integrating historical trajectories, vehicle features, environmental data, and Physical Constraints to generate accurate and diverse future vehicle trajectories.

3.1. Problem Definition

The multi-agent trajectory prediction task in autonomous driving scenarios involves forecasting the future trajectories of a dynamic set of interacting vehicles $A_1, A_2, \ldots, A_{N_t}$, where N_t represents the number of vehicles at time t, and each vehicle A_i belongs to a specific semantic class C_i (e.g., Car, Bus, Truck). Our goal is to predict their trajectories over the next T timesteps by considering their historical states.

At time t, we represent each vehicle A_i using a comprehensive state vector $s_i^t \in \mathbb{R}^D$ that includes kinematic states and external information. The external information incorporates vehicle-specific features, environmental features from HD maps, and interaction features.

The sequence of historical states over the past H timesteps is defined as

$$\mathbf{X} = s_i^{t-H+1:t} = \{s_i^{t-H+1}, s_i^{t-H+2}, \ldots, s_i^t\}$$

where $\mathbf{X}$ captures the complete movement history and contextual evolution of vehicle A_i from the past H timesteps up to the current time t.

Our task is to predict the future states of each vehicle over the next T timesteps, represented as:

$$\mathbf{Y} = r_i^{t+1:t+T} = \{r_i^{t+1}, r_i^{t+2}, \ldots, r_i^{t+T}\}$$

While $\mathbf{Y}$ could, in principle, include predictions for all features such as position, velocity, and heading, in this work, we focus solely on predicting the position information r_i^{t+T} of each vehicle.

Therefore, given the input variable $\mathbf{X}$, our goal is to model the conditional probability distribution $p(\mathbf{Y} \mid \mathbf{X})$ for the future positions $\mathbf{Y}$ of all vehicles. In this paper, this conditional probability distribution $p(\mathbf{Y} \mid \mathbf{X})$ is represented by an associated probability density function.

This distribution captures how the future positions of all vehicles are influenced by their past movement history, vehicle-specific features, environmental context, and interactions with other neighboring vehicles. Through this modeling approach, we aim to comprehensively capture the dynamic behavior of the vehicles and their complex relationships with the surrounding environment, while specifically focusing on predicting accurate future positions.

3.2. Input Feature Extraction and Encoding

Accurate trajectory prediction relies on integrating multiple modalities to comprehensively understand vehicle behavior and environmental interactions. Our model combines various input modalities, including vehicle historical trajectories, vehicle attributes, map data, and interaction features, to capture key factors influencing vehicle dynamics and provide a rich context for accurate future trajectory predictions.

3.2.1. Historical Trajectory Encoding

To capture temporal dependencies in the motion data, we use an LSTM network to encode the historical trajectory of each vehicle. For a scene at time t with N_t vehicles, the input sequence for each vehicle i is defined as

$$\mathbf{X} = s_i^{t-H+1:t} \in \mathbb{R}^{H \times D}$$

where $s_i^{t-H+1:t}$ represents the sequence of historical states for vehicle i over the past H timesteps. Each state s_i^{τ} (for $\tau = t - H + 1, \ldots, t$) includes features such as position, velocity, heading, and steering angle and has dimensionality D.

For each vehicle, we input its sequence of historical states into an LSTM network [31,48]. The LSTM processes each state in the sequence step-by-step, updating its hidden state at each time step to capture temporal dependencies in the vehicle's motion. This iterative approach enables the LSTM to retain relevant motion patterns and dependencies within the sequence. The final hidden state vector, which is a 32-dimensional vector, serves as a summary of the vehicle's motion history and is used as an input feature for the subsequent prediction model.

3.2.2. Vehicle-Specific Features

To accurately capture the motion dynamics of different vehicles, our model integrates vehicle-specific features such as type and size. We classify vehicles into six types, car, bus, truck, trailer, construction vehicle, and emergency vehicle, each represented by an embedding layer that maps the type information into dense vectors [49]. This embedding captures latent relationships between different vehicle types, allowing the model to distinguish behavior patterns unique to each type.

For physical dimensions, we apply z-score normalization to the length, width, and height of vehicles. This normalization involves subtracting the mean and dividing by the standard deviation for each dimension, resulting in features that have a mean of zero and a standard deviation of one. This transformation reduces discrepancies in scale and enhances stability during model training.

The normalized dimensions are concatenated with the vehicle type embeddings to create a unified feature vector, which encodes both physical size and categorical information about the vehicle type. This vector is then fed into a multi-layer perceptron (MLP) [50] consisting of three fully connected layers with 128, 64, and 32 neurons, respectively. Each layer uses a ReLU activation function to introduce non-linearity, and the final output is a 32-dimensional embedding that encapsulates the key characteristics of each vehicle.

We train the MLP from scratch on our dataset using the Adam optimizer, enabling it to learn complex interactions between vehicle types and dimensions. The resulting embedding captures key characteristics of each vehicle, enhancing the trajectory prediction model's ability to generate precise, context-sensitive predictions.

3.2.3. Map Feature Extraction

To incorporate environmental context into the trajectory prediction model, we use a local semantic map centered on each vehicle. The input map, M_i^t, is a binary tensor where each element can be 0 or 1, with dimensions $h \times w \times l$, where h and w represent height and width, and l indicates the number of semantic layers. Each layer corresponds to features such as the drivable area, road divider, lane divider, stop line, and pedestrian crossing.

As shown in Figure 3, the semantic map includes three separate channels: drivable areas, road dividers, and lane dividers. Additionally, a composite RGB image combines these channels to provide a comprehensive view of the local road environment around the vehicle.

To ensure spatial consistency, the map is aligned with the vehicle's heading direction. After alignment, the map patch is processed using a CNN [51] with multiple convolutional layers, which capture both broad spatial patterns and finer details.

To align the local semantic map with the vehicle's heading direction, the map is rotated to match the orientation of the vehicle's travel. In Figure 4, the left image (a) shows the original map before rotation, and the right image (b) illustrates the map after rotation. This alignment ensures spatial consistency, which is crucial for accurate trajectory prediction. The cyan square represents the vehicle, and the black arrow indicates its heading direction.

The resulting feature representation is a 32-dimensional vector that encapsulates key spatial information from the vehicle's local environment. This vector, combined with the vehicle's motion history, enables the model to generate more precise and context-aware trajectory predictions, taking into account both dynamic behavior and static road infrastructure.

Additionally, this feature extraction process can be extended to include other sensor data, such as LIDAR or camera images, further enhancing the model's understanding of complex traffic scenarios.

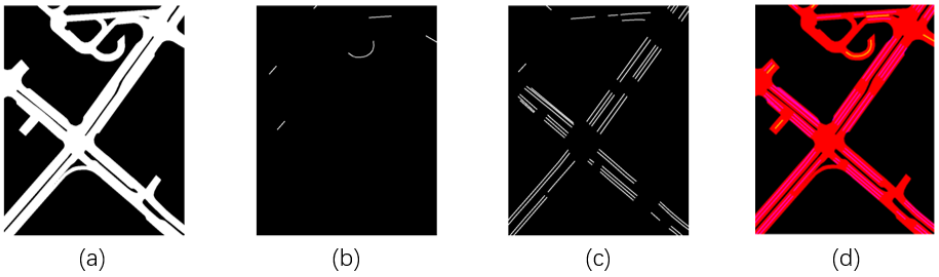

Figure 3. The semantic map includes three separate channels: (**a**) drivable areas, (**b**) road dividers, (**c**) lane dividers, and (**d**) a composite RGB image combining the three channels.

Figure 4. Comparison of mask maps before and after rotation based on the vehicle's heading direction. (**a**) Before rotation. (**b**) After rotation.

3.2.4. Interaction Network

To model the interaction of vehicles with other agents in the environment, we represent the scene as a graph, where each agent is treated as a vertex, and edges indicate potential interactions [26]. Each vertex is associated with a semantic category, such as "Car", "Bus", or "Pedestrian". A directed edge from entity E_i to E_j exists if E_i can influence E_j. This potential influence is determined by evaluating the distance between their positions, represented as

$$\|p_i - p_j\| \leq d_{\text{interaction}}$$

where p_i and p_j are the 2D coordinates of entities E_i and E_j, and $d_{\text{interaction}}$ is a threshold distance based on the interaction type between entities E_i and E_j (e.g., "car-pedestrian", "car-car").

The interaction network represents the relative relationship between the current agent and its neighboring agents using edge features. For the same type of edge (e.g., "car-car"), the model aggregates these edge features using element-wise summation. This aggregation

method enables the model to flexibly handle varying numbers of neighboring entities while preserving the interaction information conveyed by each edge type. The aggregated edge features for each type are then fed into an LSTM with shared weights across all connections of the same type (e.g., "car-pedestrian"), which encodes the influence exerted by neighboring entities over time.

To capture the combined influence of different types of edges on the target entity, the model employs an additive attention mechanism [52]. This mechanism assigns dynamic weights to each edge type, allowing the model to emphasize the most relevant interactions for the target agent within a given context. Specifically, the attention score e_{ij} between the target agent's state q_i and each edge type k_j is calculated as follows:

$$e_{ij} = v^\top \tanh(W_q q_i + W_k k_j)$$

where q_i is the encoded feature of the target agent, k_j represents the LSTM-encoded feature for each edge type, and W_q, W_k, and v are learnable parameters. The scores are then normalized with a Softmax function to produce attention weights α_{ij}:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{j'=1}^{M} \exp(e_{ij'})}$$

where M is the total number of neighbor types. Finally, the model computes a weighted sum of the edge-type encodings, yielding the final influence representation for the target agent:

$$\text{Influence} = \sum_{j=1}^{M} \alpha_{ij} k_j$$

This attention mechanism ensures that the model focuses on the most significant interactions, producing a 32-dimensional weighted representation of the overall influence that neighboring agents exert on the target agent. By dynamically adjusting these weights based on specific interactions, the model gains a comprehensive understanding of the relationships among multiple entities, enhancing its performance in complex, multi-agent scenarios.

3.3. Prediction Module

The prediction module is used to forecast future trajectories of multiple interacting agents in complex driving environments. It consists of two main components: Latent Variable Modeling and a Decoder with Physical Constraints. In the following sections, we provide a detailed explanation of each component and its role in generating robust and context-aware predictions.

3.3.1. Latent Variable Modeling

To effectively capture the multimodal nature of future trajectories, the prediction module employs an Information-Maximizing Variational Autoencoder (infoVAE) framework [53]. Additionally, we introduce a high-level latent variable z, which represents a finite set of possible high-level behaviors [26]. The latent variable z encodes these behaviors, enabling the model to represent multiple possible future outcomes. The overall conditional probability distribution of future trajectories $p(\mathbf{Y} \mid \mathbf{X})$ is expressed by marginalizing over the latent variable:

$$p(\mathbf{Y} \mid \mathbf{X}) = \sum_{z} p_{\theta_2}(\mathbf{Y} \mid \mathbf{X}, z)\, p_{\theta_1}(z \mid \mathbf{X})$$

where $\mathbf{X} \in \mathbb{R}^n$ is the encoded context vector, including historical trajectory, vehicle features, interaction features, and environmental information, with $n = 128$; $\mathbf{Y} \in \mathbb{R}^m$ represents the predicted future trajectory, with $m = 2$; and z is the set of discrete latent variables, consisting of 20 discrete elements. Here, $p_{\theta_1}(z \mid \mathbf{X})$ and $p_{\theta_2}(\mathbf{Y} \mid \mathbf{X}, z)$ are parameterized by θ_1 and θ_2, respectively, representing conditional probability distributions.

Training Loss Function

In the infoVAE framework [53], a mutual information (MI) term is introduced in the loss function by maximizing the correlation between the latent variable z and the input context $\mathbf{x}$. The MI term ensures that the latent variable z effectively captures the diverse modalities of future behavior given the context.

The objective of training the model is to minimize the following loss function:

$$\mathcal{L} = -\lambda \cdot D_{\mathrm{KL}}\left(q_\phi(z \mid \mathbf{X}, \mathbf{Y}) \parallel p_{\theta_1}(z \mid \mathbf{X})\right) - \mathbb{E}_{q_\phi(z|\mathbf{X},\mathbf{Y})}\left[\log p_{\theta_2}(\mathbf{Y} \mid \mathbf{X}, z)\right] + \alpha \cdot I(\mathbf{X}; z)$$

where the first term $-\lambda \cdot D_{\mathrm{KL}}\left(q_\phi(z \mid \mathbf{X}, \mathbf{Y}) \parallel p_{\theta_1}(z \mid \mathbf{X})\right)$ is the Kullback–Leibler divergence term, which regularizes the posterior distribution to be close to the prior distribution, preventing overfitting and encouraging smoothness in the latent space; the second term $-\mathbb{E}_{q_\phi(z|\mathbf{X},\mathbf{Y})}\left[\log p_{\theta_2}(\mathbf{Y} \mid \mathbf{X}, z)\right]$ is the reconstruction loss, ensuring that the predicted trajectory is close to the ground truth trajectory; and the third term $\alpha \cdot I(\mathbf{X}; z)$ is the mutual information term, which maximizes the dependency between the context $\mathbf{x}$ and the latent variable z.

The mutual information $I(\mathbf{X}; z)$ [54] is defined as

$$I(\mathbf{X}; z) = \mathbb{E}_{p_{\theta_1}(\mathbf{X}, z)}\left[\log \frac{p_{\theta_1}(z \mid \mathbf{X})}{p_{\theta_1}(z)}\right]$$

where the unconditional latent distribution $p_{\theta_1}(z)$ is obtained by averaging over all $\mathbf{x}$ in the batch.

Model Training

During model training, the encoder receives the input information $\mathbf{x}$ and the corresponding ground truth future trajectory $\mathbf{y}$, generating the posterior distribution $q_\phi(z \mid \mathbf{X}, \mathbf{Y})$ [55]. The encoder and decoder networks are optimized to minimize the total loss function $\mathcal{L}$. The inclusion of the mutual information term increases the interpretability and robustness of the latent representations, enabling the model to generate diverse and contextually appropriate trajectories for each agent. By balancing reconstruction accuracy, latent space regularization, and informativeness of the latent variables, the infoVAE framework generates a comprehensive set of possible future outcomes given the current context.

3.3.2. Decoder with Physical Constraints

The decoder generates future trajectories that are diverse and dynamically feasible by incorporating vehicle kinematic constraints. The prediction process involves three main components, a GRU module, a bivariate Gaussian distribution, and a bicycle model, which work together in sequence to generate physically feasible trajectories.

A. GRU Module for State Prediction

The first component of the decoder is a GRU [56] module that processes the encoded information. The GRU module predicts a sequence of control actions over the specified prediction horizon. It takes as input the latent variable z sampled from the latent space and the encoded context vector $\mathbf{x}$. At each time step t, the GRU outputs the parameters of a bivariate Gaussian distribution for control actions $\mathbf{u}^t = [a_k^t, \delta_k^t]$, where a_k^t is the acceleration and δ_k^t is the steering angle. These parameters are then used to define the distribution of possible control actions.

B. Bivariate Gaussian Distribution for Control Modeling

To model the uncertainty in the GRU's predictions, we employ a bivariate Gaussian distribution to capture the joint distribution of possible control actions. This approach allows us to represent the variability in acceleration and steering angle across future time steps, providing a probabilistic framework that enables the generation of diverse and feasible trajectories.

In this context, a Gaussian distribution refers to a probability distribution that is fully described by its mean and covariance matrix. For each time step t, the GRU outputs parameters for a bivariate Gaussian distribution that defines the probability distribution of control action variables $\mathbf{U}$ [57]. The probability distribution of control actions at each time step t is given by

$$p(\mathbf{U} \mid \mathbf{X}, z) = \mathcal{N}(\mathbf{u}^t \mid \mu^t, \Sigma_u^t)$$

where $\mu^t = [\mu_a^t, \mu_\delta^t]$ is the mean vector, representing the expected values of acceleration a_k^t and steering angle δ_k^t, and Σ_u^t is the covariance matrix that captures the uncertainties and correlations between these control actions:

$$\Sigma_u^t = \begin{bmatrix} (\sigma_a^t)^2 & \rho_{a\delta}^t \, \sigma_a^t \, \sigma_\delta^t \\ \rho_{a\delta}^t \, \sigma_a^t \, \sigma_\delta^t & (\sigma_\delta^t)^2 \end{bmatrix}$$

Here, $(\sigma_a^t)^2$ and $(\sigma_\delta^t)^2$ denote the variances of acceleration and steering angle at time t, respectively, while $\rho_{a\delta}^t$ represents the correlation between them. This covariance structure captures the range of possible control actions and their interdependencies, allowing for a flexible representation of uncertainties in control at each step.

By using this probabilistic model, we can sample control actions $\mathbf{u}^t$ from this distribution at each time step. These sampled actions provide a range of dynamically feasible trajectories when combined with the Physical Constraints imposed by the kinematic model.

C. Bicycle Model for Trajectory Generation

Finally, to ensure physical feasibility, the control actions sampled from this bivariate Gaussian distribution are integrated with the vehicle's kinematic model. We employ the bicycle model [58], a simplified representation of vehicle dynamics, to transform these control actions into vehicle trajectories in the position space. The equations governing the bicycle model are given as follows:

$$\begin{aligned} \dot{x}_k^t &= v^t \cos(\theta_k^t) \\ \dot{y}_k^t &= v^t \sin(\theta_k^t) \\ \dot{\theta}_k{}^t &= \frac{v_k^t}{L} \tan(\delta_k^t) \\ \dot{v}_k^t &= a_k^t \end{aligned}$$

where (x_k^t, y_k^t) are the coordinates of the vehicle in the 2D plane at time t, θ_k^t is the vehicle's heading angle, v_k^t is the vehicle's speed, δ_k^t is the steering angle, a_k^t is the acceleration, and L is the wheelbase of the vehicle. These equations describe the evolution of the vehicle's position (x_k^t, y_k^t) and the rate of change in heading over time under the influence of the control actions a_k^t and δ_k^t.

4. Experiments

4.1. Datasets

For this work, we utilize the nuScenes dataset [59], a large-scale dataset designed for autonomous driving research. nuScenes provides a comprehensive set of multi-modal sensor data, high-definition maps, and detailed annotations, making it an ideal resource for trajectory prediction tasks.

The dataset comprises 1000 scenes, each 20 s long, recorded in diverse urban environments across Boston and Singapore. The data are captured at a frequency of 2 Hz using multiple sensors, including LiDAR, radars, and cameras, providing synchronized, high-resolution information about the environment. Annotations are available for various types of dynamic agents (e.g., cars, buses, trucks, and pedestrians) and static elements such as road infrastructure.

In addition to the dynamic agent data, nuScenes offers high-definition semantic maps, which include detailed information (e.g., lane boundaries, drivable areas, road dividers, and crosswalks). These maps are integral to understanding the environment in which the agents operate and are crucial for tasks involving trajectory prediction in complex urban scenarios.

For our trajectory prediction task, we extract historical trajectories up to 2 s as input features, along with corresponding local map patches. The future trajectories of 6 s serve as ground truth for training and evaluation. We adhere to the official nuScenes data split, using 700 scenes for training, 150 for validation, and 150 for testing.

4.2. Implementation Details

The model was implemented using PyTorch and trained on an NVIDIA RTX A6000 GPU with 48 GB of memory for 20 epochs. A batch size of 512 was used with the Adam optimizer and an initial learning rate of 0.002, decaying exponentially by 0.9999 per epoch. Gradient clipping with a cap of 1.0 was applied to ensure training stability.

The dataset was preprocessed by transforming agent trajectories into a local coordinate system, setting each agent's initial position at the origin and aligning their heading.

The model employed a CVAE structure, which incorporated both KL divergence and MI terms. The KL weight started at 0 and gradually increased to 100 using a sigmoid schedule, allowing the model to prioritize reconstruction early in training and later focus on regularizing the latent space. The MI term was fixed at $\alpha = 1$, ensuring the latent variable z captured relevant information from the input $\mathbf{x}$, balancing multimodality and latent space structure.

The model was evaluated on 150 test scenes from the nuScenes dataset. Multiple plausible trajectories were generated by sampling from GMM, accounting for uncertainty in agent behavior and ensuring realistic trajectory predictions.

For a more detailed discussion on the model's runtime performance during online inference, including metrics such as inference time, frame rate, and memory usage, please refer to Appendix A.

4.3. Evaluation Metrics

To evaluate the performance of the proposed trajectory prediction model, as in previous work [26,27,32–34,60–62], we used the following key metrics:

Minimum Average Displacement Error (minADE)

minADE measures the minimum average Euclidean distance between the predicted trajectory and the ground truth trajectory across all time steps among the predicted trajectories. It primarily evaluates the prediction accuracy of the model across the entire trajectory:

$$\text{minADE}_k = \min_{i=1,\ldots,k} \left(\frac{1}{T} \sum_{t=1}^{T} \left\| y_t^{i,\text{pred}} - y_t^{\text{true}} \right\|_2 \right)$$

where i is the index of a sampled predicted trajectory from the set of generated trajectories, and T is the prediction horizon. Here, $\| \cdot \|_2$ denotes the Euclidean (L2) norm, which is used to measure the Euclidean distance between the predicted and true trajectories.

Minimum Final Displacement Error (minFDE)

minFDE measures the minimum Euclidean distance between the predicted and true positions at the final time step among the predicted trajectories. This metric focuses on the final prediction accuracy, especially at the last predicted point:

$$\text{minFDE}_k = \min_{i=1,\ldots,k} \left(\left\| y_T^{i,\text{pred}} - y_T^{\text{true}} \right\|_2 \right)$$

where i refers to the index of the closest predicted trajectory from the set.

Miss Rate at Distance d ($\text{MissRate}_{k,d}$)

$\text{MissRate}_{k,d}$ measures the proportion of predicted trajectories where the maximum pointwise L2 distance exceeds a threshold d. If any time step's error is greater than d, the trajectory is considered a miss. This metric evaluates the model's ability to predict trajectories within a certain tolerance, assessing its prediction accuracy. It is defined as

$$\text{MissRate}_{k,d} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}\left[\min_{j=1,\ldots,k}\left(\max_{t=1,\ldots,T}\left\|y_t^{j,\text{pred}} - y_t^{\text{true}}\right\|_2\right) > d\right]$$

where k refers to the top k predicted trajectories, d is the threshold, and N is the total number of agents. In our case, $d = 2$ m, a threshold commonly used in the nuScenes benchmark for trajectory prediction tasks. This value is chosen to reflect a balance between precision and safety in autonomous driving, where predicting within a 2 m margin is crucial for maintaining safe distances from other vehicles and obstacles in complex driving environments.

Off-Road Rate

Off-Road Rate evaluates the percentage of predicted trajectories that leave the drivable area. This metric is used to assess the model's adherence to driving constraints, ensuring that the predicted trajectories stay within the drivable area:

$$\text{Off-Road Rate} = \frac{\text{Number of off-road trajectories}}{\text{Total number of trajectories}}$$

Kernel Density Estimate Negative Log-Likelihood (KDE NLL)

KDE NLL measures the quality of the predicted probability distribution by evaluating the likelihood of the ground truth trajectory under a kernel density estimate formed from the predicted trajectories. This metric captures the overall quality of the model's multimodal behavior and uncertainty:

$$\text{KDE NLL}_k = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{K}\sum_{k=1}^{K}\mathcal{N}(y_t^{\text{true}} \mid y_t^{k,\text{pred}}, \Sigma_p)\right)$$

where $\mathcal{N}(\cdot \mid y_t^{k,\text{pred}}, \Sigma_p)$ represents a Gaussian distribution centered at the predicted point $y_t^{k,\text{pred}}$ with covariance Σ_p, K is the number of predicted trajectories, and N is the number of agents.

These metrics comprehensively evaluate the model's prediction accuracy, multimodal behavior coverage, and adherence to driving constraints, providing a thorough assessment of its performance in real-world scenarios.

4.4. Results and Analysis

4.4.1. Quantitative Results

We evaluate our proposed method against several state-of-the-art baseline models on the nuScenes dataset. Table 1 presents a comprehensive comparison of our method with these baselines across various metrics. The subscripts 1, 5, 10 in MinADE, MinFDE, and MissRate indicate that the metric is calculated by selecting the best trajectory from the top 1, 5, or 10 predicted trajectories.

Our proposed MTP-HPC method demonstrates strong performance on all evaluation metrics. For single-trajectory prediction, our method achieves a MinADE_1 of 2.14 and a MinFDE_1 of 5.03, indicating high accuracy in predicting the most probable future path of vehicles.

Table 1. Comparison with baseline models on the nuScenes dataset (units: meters for minADE, minFDE; percentage for MissRate and Off-Road Rate).

	MinADE_1 (m)	MinADE_5 (m)	MinADE_{10} (m)	MinFDE_1 (m)	MinFDE_5 (m)	MinFDE_{10} (m)	$\text{MissRate}_{5,2}$ (%)	$\text{MissRate}_{10,2}$ (%)	Off-Road Rate (%)
Const vel and yaw	4.61	4.61	4.61	11.21	11.21	11.21	91	91	14
Physics oracle	3.69	3.69	3.69	9.06	9.06	9.06	88	88	12
MTP [60]	4.42	2.22	1.74	10.36	4.83	3.54	74	67	25
Multipath [61]	4.43	1.78	1.55	10.16	3.62	2.93	78	76	36
CoverNet [62]	-	2.62	1.92	11.36	-	-	76	64	13
Trajectron++ [26]	-	1.88	1.51	9.52	-	-	70	57	25
MHA-JAM [32]	3.77	1.85	1.24	8.65	3.85	2.23	59	45	7
PGP [33]	-	1.27	0.94	7.17	-	-	52	34	3
FRM [27]	-	1.18	**0.88**	6.59	-	-	48	30	2
CASPNet++ [34]	2.74	**1.18**	0.93	6.19	-	-	50	**30**	**1**
Our Method (MTP-HPC)	**2.14**	1.26	0.99	**5.03**	**2.85**	**2.16**	**41**	32	2

In multiple trajectory prediction scenarios, our method also shows robust performance with MinADE_5 and MinADE_{10} values of 1.26 and 0.99, respectively. Notably, our method achieves the best performance in MinFDE_5 (2.85) and MinFDE_{10} (2.16) among all compared methods, highlighting its effectiveness in long-term trajectory prediction.

The low $\text{MissRate}_{5,2}$ of 0.41 demonstrates our model's ability to generate accurate predictions within a 2 m threshold, which is crucial for safety in autonomous driving applications. Furthermore, our model achieves a low Off-Road Rate of 0.02, showcasing its effectiveness in predicting trajectories that adhere to road constraints.

To further analyze our model's performance, we examine how different metrics change over various prediction horizons. Figure 5 illustrates the average values from 1 to 6 s.

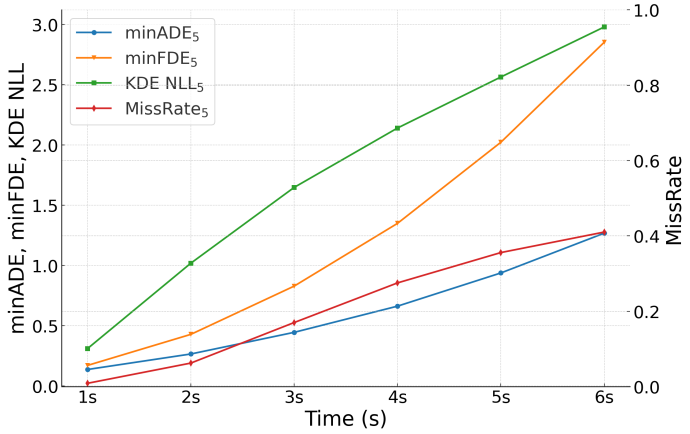

Figure 5. Trajectory prediction metrics over different prediction horizons for all vehicles.

As expected, prediction errors generally increase with longer time horizons. The minADE_5 and minFDE_5 metrics show a steady increase, reflecting the growing uncertainty in predicting vehicle positions further into the future. The KDE NLL_5 metric also increases over time, indicating that the model's probabilistic predictions become less certain for longer-term forecasts.

To evaluate the versatility of our model, we analyze its performance across different vehicle types. Table 2 presents the minADE_5, minFDE_5, and KDE NLL $_5$ metrics for various vehicle types at 2, 4, and 6 s prediction horizons. Figure 6 illustrates the performance metrics for different vehicle types when considering 5, 10, and 15 predicted trajectories.

Our model demonstrates good performance across the majority of vehicle types. Analysis results show that the model can consistently generate accurate trajectory predictions for all evaluated vehicle types.

Table 2. Trajectory prediction performance across vehicle types and time horizons.

Type	minADE_5			minFDE_5			KDE NLL_5		
	2 s	4 s	6 s	2 s	4 s	6 s	2 s	4 s	6 s
BUS	0.23	0.79	1.74	0.31	1.58	3.94	1.29	3.25	4.70
CAR	0.18	0.60	1.30	0.23	1.18	2.98	0.17	1.82	2.98
TRAILER	0.17	0.54	1.07	0.22	1.00	2.31	−0.10	2.30	3.59
CONSTRUCTION	0.05	0.13	0.20	0.06	0.19	0.32	−2.66	−1.09	−0.40
EMERGENCY	0.51	2.58	5.00	0.69	5.26	10.09	4.47	8.52	10.38
TRUCK	0.18	0.59	1.26	0.24	1.14	2.80	−0.03	1.90	3.08

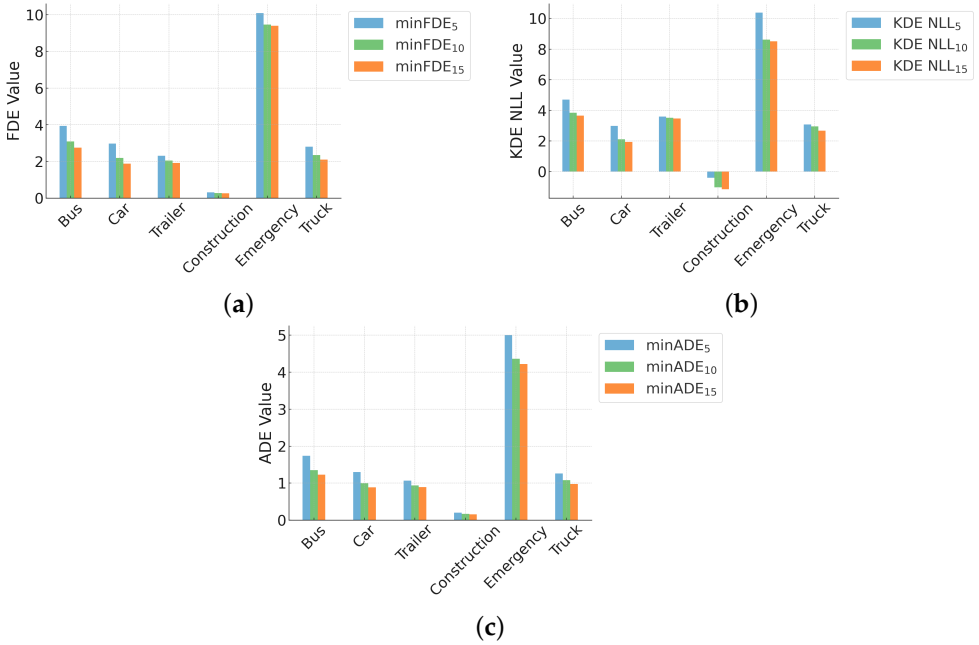

Figure 6. Performance metrics across vehicle types for varying numbers of predicted trajectories: (**a**) minFDE, (**b**) KDE NLL, and (**c**) minADE for 5, 10, and 15 predictions.

Notably, construction vehicles exhibit the lowest error rates across all metrics and time horizons. On the other hand, emergency vehicles show relatively higher error rates, primarily due to the limited number of samples for this vehicle type in the dataset.

For vehicle types that occupy the majority of road traffic, such as cars, trucks, buses, and trailers, our model demonstrates stable and reliable prediction capabilities. These results highlight the applicability and effectiveness of our method in real-world traffic scenarios.

The KDE NLL metric further confirms the excellent performance of our model across different vehicle types, with most vehicle types achieving low KDE NLL values. This suggests that our model produces accurate and reliable probability distributions for trajectory predictions across various vehicle categories.

4.4.2. Qualitative Analysis

To visually evaluate our model's performance, we conducted a visualization analysis of trajectory predictions for different vehicle types. Figure 7 shows the predicted trajectories for four typical vehicle types: bus, car, trailer, and truck.

From the figure, we can observe that our model is capable of generating diverse trajectory predictions for different types of vehicles. The predicted trajectories for each vehicle type exhibit a range of distributions, reflecting the model's multiple predictions for possible future paths.

These visualization results demonstrate the consistency of our model in handling different vehicle types. Whether for large vehicles (such as buses and trailers) or smaller vehicles

(like cars), the model is able to generate a series of possible trajectories. The distribution of predicted trajectories reflects the model's estimation of uncertainty in future positions.

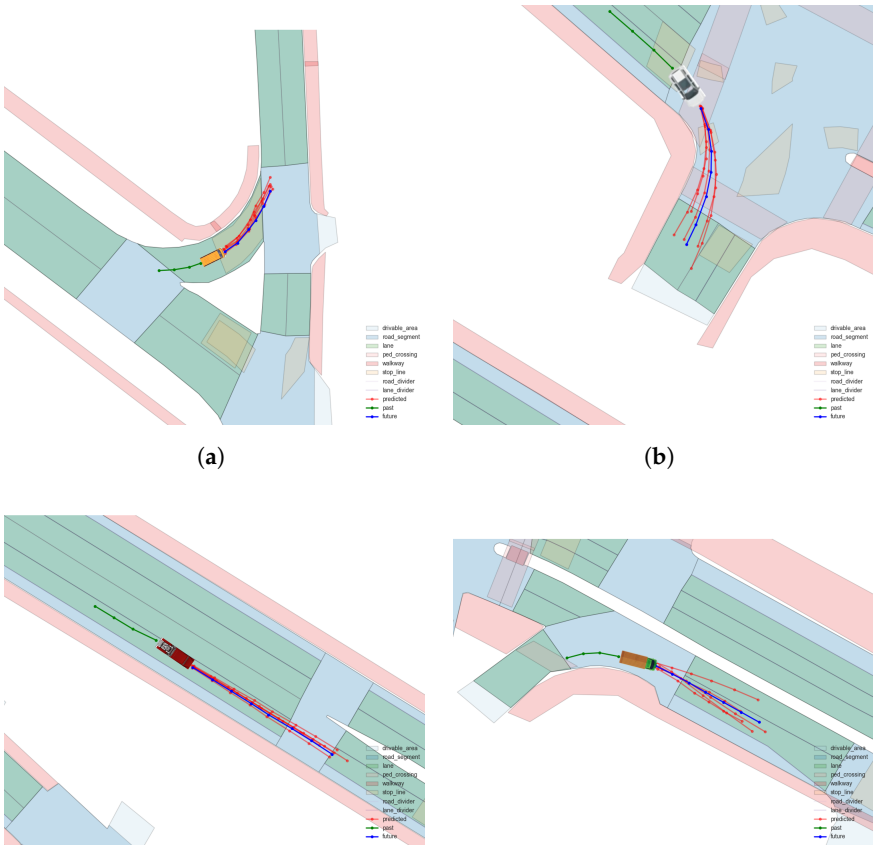

Figure 7. Trajectories of different vehicle types. (**a**) Bus trajectory. (**b**) Car trajectory. (**c**) Trailer trajectory. (**d**) Truck trajectory.

4.4.3. Ablation Studies

To understand the contribution of each component in our model, we conducted a series of ablation studies. Table 3 presents the results of these studies, showing how different combinations of components affect the model's performance.

Table 3. Impact of model components on prediction performance.

Base	Map	Physical Constraint	Vehicle Feature	minADE_5	minFDE_5	KDE NLL_5
✓	×	×	×	1.76	4.02	3.36
✓	✓	×	×	1.41	3.13	3.29
✓	×	✓	×	1.46	3.29	3.16
✓	×	×	✓	1.42	3.07	3.02
✓	✓	✓	×	1.29	2.90	3.03
✓	×	✓	✓	1.32	2.99	2.91
✓	✓	×	✓	1.37	3.04	3.23
✓	✓	✓	✓	**1.26**	**2.85**	**2.89**

The ablation studies investigate the impact of three key components: map information, Physical Constraints, and vehicle features. From the results, we can observe that the base model, without any additional components, shows the highest error rates across all metrics. Each component, when added individually to the base model, improves performance.

The combination of all three components yields the best performance across all metrics, with a minADE_5 of 1.26, a minFDE_5 of 2.85, and a KDE NLL_5 of 2.89. The Physical Constraint and vehicle feature components, when combined with map information, show synergistic effects, further reducing error rates.

These results demonstrate the importance of each component in our model architecture. The map information provides crucial context about the environment, the Physical Constraints ensure realistic predictions, and the vehicle features allow the model to adapt to different vehicle types. The synergistic effect of these components results in a model that outperforms simpler variants across all evaluated metrics.

5. Conclusions

In this work, we proposed MTP-HPC, a novel multi-agent trajectory prediction model that effectively integrates multiple input modalities, including historical trajectory data, interaction features, and environmental context. By leveraging a CVAE framework along with a physically constrained decoder, our model captures the multimodal nature of future outcomes, generating diverse yet dynamically feasible trajectory predictions. This multimodal integration enables the model to handle complex real-world driving environments, accurately predicting future trajectories for various vehicle types, such as cars, trucks, and emergency vehicles.

The performance evaluation on the nuScenes dataset demonstrates significant improvements in both accuracy and realism. Our model achieves great results in key metrics such as minADE and minFDE, reflecting its ability to generate precise short- and long-term predictions. The low Off-Road Rate further underscores the model's effectiveness in producing safe and realistic trajectories that comply with road constraints. Additionally, the competitive KDE NLL scores highlight the model's ability to manage uncertainty and generate multiple plausible future paths, which is crucial for dynamic urban driving scenarios.

Despite these strong results, there are areas for future improvement. Currently, our model uses a CNN architecture for map feature processing and interaction modeling due to its stability in spatial feature extraction and computational efficiency. In future work, we plan to explore more recent network architectures, such as Vision Transformers and Graph Transformer Networks, aiming to further improve model performance while balancing prediction accuracy and computational efficiency. Additionally, as prediction time horizons increase, our model's prediction error also grows, indicating the need for better methods to capture long-term dependencies. Accurate long-term predictions are essential for decision-making in autonomous driving. Furthermore, the model's higher error rates for rare vehicle types, such as emergency vehicles, suggest the need for further refinement through targeted data augmentation or specialized training to improve generalizability across diverse agent types.

In future work, we plan to evaluate the model on additional datasets to further assess its generalization capabilities across different environments and driving conditions. We will also explore online deployment, enabling the model to operate in real time and continuously adapt to evolving traffic scenarios.

Author Contributions: Conceptualization, M.G. and M.D.; methodology, M.G.; software, M.G.; validation, M.D., K.O. and Y.N.; formal analysis, M.G.; writing—original draft preparation, M.G.; writing—review and editing, M.G., K.O., Y.N. and Y.Z.; visualization, M.G.; supervision, K.O. and K.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Nagoya University and JSPS KAKENHI Grant Number 24K20837.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in this study are publicly available as part of the nuScenes dataset, which can be accessed at https://www.nuscenes.org.

Conflicts of Interest: The author Ming Ding is employed by Zhejiang Fubang Technology Inc., the author Yuxiao Zhang is employed by RoboSense Technology Co., Ltd. and the author Kazuya Takeda is employed by Tier IV Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as potential conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CVAE	Conditional Variational Autoencoder
RNN	Recurrent Neural Networks
GAN	Generative Adversarial Networks
GNN	Graph Neural Networks
LSTM	Long Short-Term Memory
MLP	Multi-Layer Perceptron
CNN	Convolutional Neural Networks
FC	Fully Connected (Layer)
GRU	Gated Recurrent Unit
GMM	Gaussian Mixture Model
minADE	Minimum Average Displacement Error
minFDE	Minimum Final Displacement Error
KDE NLL	Kernel Density Estimation Negative Log-Likelihood

Appendix A. Online Runtime Performance

In the online inference process, we conducted a detailed analysis and measurement of the model's runtime performance, covering inference time, frame rate (FPS), hardware environment, and memory usage. The experimental results indicate that the inference time and frame rate of the model on a CPU are closely related to the number of nodes and edges. An increase in the number of nodes and edges significantly increases the computational load, resulting in longer inference time and reduced frame rate.

During the experiment, the model was run on a CPU, with a model loading time of 0.03 s. The average inference time per timestep was approximately 0.09 s, corresponding to an average frame rate of 10.76 Hz. The inference time fluctuated with the number of nodes and edges, where a larger number of nodes and edges (e.g., 6–7 nodes and 40–50 edges) increased the inference time per timestep to 0.13 s, reducing the frame rate to around 7.7 Hz. Conversely, when there were fewer nodes and edges (e.g., 2–3 nodes and 20–30 edges), the inference time decreased to below 0.05 s, and the frame rate increased to 21 Hz or higher. This dynamic variation reflects the model's computational complexity changing with input size. As shown in Figure A1, we use a 3D scatter plot to visually illustrate the relationship between inference time, the number of nodes, and the number of edges. Regarding memory usage, the CPU memory usage during online inference was approximately 25 GB.

Our approach can incrementally update new observation information without completely re-executing the forward pass. This is possible due to the use of an LSTM network structure, which allows new observational data to be directly input into the last LSTM unit of the encoder, effectively updating the model representation without repeating the entire inference process.

During inference, we chose to use a CPU to simulate the conditions of a real deployment environment. In many real-world applications, especially on resource-constrained edge devices, GPUs may not always be available. Additionally, for cost and energy efficiency, CPUs are often preferred. We tested inference on a CPU to ensure that the model maintains high real-time performance and stability under limited computational resources. This also demonstrates the model's adaptability and potential applicability in low-resource environments.

The current CPU inference frame rate is 10–11 Hz, which meets the needs of some medium-level real-time applications, overall, the performance of online inference is as

expected, and the model can adapt to changes in the number of nodes and edges in different scenarios.

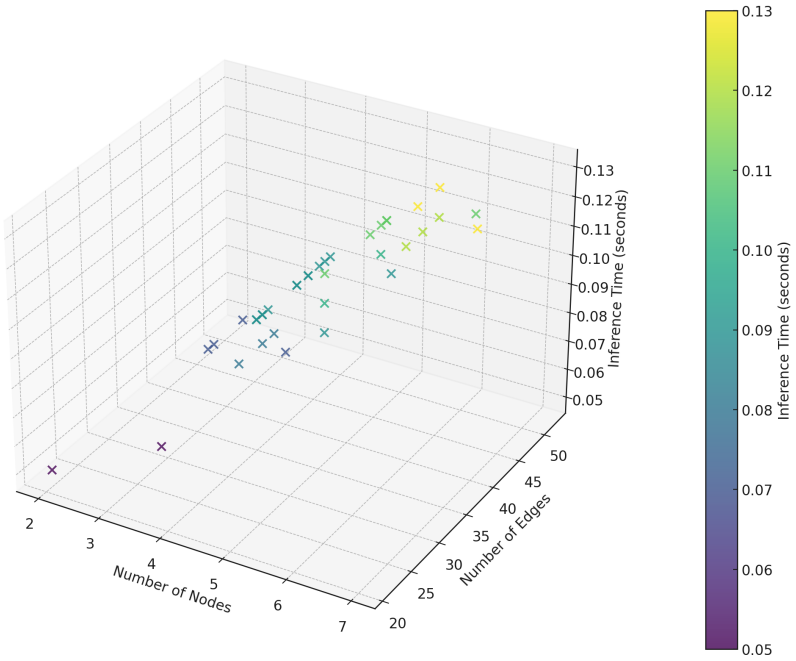

Figure A1. Three-dimensional scatter plot illustrating the relationship between inference time, the number of nodes, and the number of edges in online inference.

References

1. Lefèvre, S.; Vasquez, D.; Laugier, C. A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH J.* **2014**, *1*, 1. [CrossRef]
2. Zhan, W.; Sun, L.; Wang, D.; Shi, H.; Clausse, A.; Naumann, M.; Kummerle, J.; Konigshof, H.; Stiller, C.; de La Fortelle, A.; et al. INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv* **2019**, arXiv:1910.03088.
3. Rudenko, A.; Palmieri, L.; Herman, M.; Kitani, K.M.; Gavrila, D.M.; Arras, K.O. Human motion trajectory prediction: A survey. *Int. J. Robot. Res.* **2020**, *39*, 895–935. [CrossRef]
4. Koopman, P.; Wagner, M. Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intell. Transp. Syst. Mag.* **2017**, *9*, 90–96. [CrossRef]
5. Markkula, G.; Lin, Y.S.; Srinivasan, A.R.; Billington, J.; Leonetti, M.; Kalantari, A.H.; Yang, Y.; Lee, Y.M.; Madigan, R.; Merat, N. Explaining human interactions on the road by large-scale integration of computational psychological theory. *PNAS Nexus* **2023**, *2*, pgad163. [CrossRef]
6. Chandra, R.; Bera, A.; Manocha, D. Stylepredict: Machine theory of mind for human driver behavior from trajectories. *arXiv* **2020**, arXiv:2011.04816.
7. Sadigh, D.; Sastry, S.; Seshia, S.A.; Dragan, A.D. Planning for autonomous cars that leverage effects on human actions. In Proceedings of the Robotics: Science and Systems, Ann Arbor, MI, USA, 18–22 June 2016; Volume 2, pp. 1–9.
8. Alahi, A.; Goel, K.; Ramanathan, V.; Robicquet, A.; Li, F.-F.; Savarese, S. Social lstm: Human trajectory prediction in crowded spaces. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 961–971.
9. Gupta, A.; Johnson, J.; Fei-Fei, L.; Savarese, S.; Alahi, A. Social gan: Socially acceptable trajectories with generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2255–2264.
10. Dinneweth, J.; Boubezoul, A.; Mandiau, R.; Espié, S. Multi-agent reinforcement learning for autonomous vehicles: A survey. *Auton. Intell. Syst.* **2022**, *2*, 27. [CrossRef]
11. Huang, Y.; Du, J.; Yang, Z.; Zhou, Z.; Zhang, L.; Chen, H. A survey on trajectory-prediction methods for autonomous driving. *IEEE Trans. Intell. Veh.* **2022**, *7*, 652–674. [CrossRef]
12. Leon, F.; Gavrilescu, M. A review of tracking and trajectory prediction methods for autonomous driving. *Mathematics* **2021**, *9*, 660. [CrossRef]
13. Schubert, R.; Richter, E.; Wanielik, G. Comparison and evaluation of advanced motion models for vehicle tracking. In Proceedings of the 2008 IEEE 11th International Conference on Information Fusion, Cologne, Germany, 30 June–3 July 2008; pp. 1–6.

14. Ammoun, S.; Nashashibi, F. Real time trajectory prediction for collision risk estimation between vehicles. In Proceedings of the 2009 IEEE 5th International Conference on Intelligent Computer Communication and Processing, Cluj-Napoca, Romania, 27–29 August 2009; pp. 417–422.
15. Polychronopoulos, A.; Tsogas, M.; Amditis, A.J.; Andreone, L. Sensor fusion for predicting vehicles' path for collision avoidance systems. *IEEE Trans. Intell. Transp. Syst.* **2007**, *8*, 549–562. [CrossRef]
16. Xiao, W.; Mehdipour, N.; Collin, A.; Bin-Nun, A.Y.; Frazzoli, E.; Tebbens, R.D.; Belta, C. Rule-based optimal control for autonomous driving. In Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems, Nashville, TN, USA, 19–21 May 2021; pp. 143–154.
17. Likmeta, A.; Metelli, A.M.; Tirinzoni, A.; Giol, R.; Restelli, M.; Romano, D. Combining reinforcement learning with rule-based controllers for transparent and general decision-making in autonomous driving. *Robot. Auton. Syst.* **2020**, *131*, 103568. [CrossRef]
18. Huang, R.; Zhuo, G.; Xiong, L.; Lu, S.; Tian, W. A Review of Deep Learning-Based Vehicle Motion Prediction for Autonomous Driving. *Sustainability* **2023**, *15*, 14716. [CrossRef]
19. Liu, J.; Mao, X.; Fang, Y.; Zhu, D.; Meng, M.Q.H. A survey on deep-learning approaches for vehicle trajectory prediction in autonomous driving. In Proceedings of the 2021 IEEE International Conference on Robotics and Biomimetics (ROBIO), Sanya, China, 27–31 December 2021; pp. 978–985.
20. Li, J.; Dai, B.; Li, X.; Xu, X.; Liu, D. A dynamic Bayesian network for vehicle maneuver prediction in highway driving scenarios: Framework and verification. *Electronics* **2019**, *8*, 40. [CrossRef]
21. Jiang, Y.; Zhu, B.; Yang, S.; Zhao, J.; Deng, W. Vehicle trajectory prediction considering driver uncertainty and vehicle dynamics based on dynamic bayesian network. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *53*, 689–703. [CrossRef]
22. Liu, J.; Luo, Y.; Zhong, Z.; Li, K.; Huang, H.; Xiong, H. A probabilistic architecture of long-term vehicle trajectory prediction for autonomous driving. *Engineering* **2022**, *19*, 228–239. [CrossRef]
23. Deo, N.; Trivedi, M.M. Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; pp. 1179–1184.
24. Park, S.H.; Kim, B.; Kang, C.M.; Chung, C.C.; Choi, J.W. Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture. In Proceedings of the 2018 IEEE Intelligent Vehicles Symposium (IV), Suzhou, China, 26–30 June 2018; pp. 1672–1678.
25. Tang, C.; Salakhutdinov, R.R. Multiple futures prediction. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2019; Volume 32.
26. Salzmann, T.; Ivanovic, B.; Chakravarty, P.; Pavone, M. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part XVIII 16; Springer: Berlin/Heidelberg, Germany, 2020; pp. 683–700.
27. Park, D.; Ryu, H.; Yang, Y.; Cho, J.; Kim, J.; Yoon, K.J. Leveraging future relationship reasoning for vehicle trajectory prediction. *arXiv* **2023**, arXiv:2305.14715.
28. Li, J.; Ma, H.; Zhan, W.; Tomizuka, M. Generic probabilistic interactive situation recognition and prediction: From virtual to real. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 3218–3224.
29. Zyner, A.; Worrall, S.; Nebot, E. A Recurrent Neural Network solution for predicting driver intention at unsignalized intersections. *IEEE Robot. Autom. Lett.* **2018**, *3*, 1759–1764. [CrossRef]
30. Sutskever, I. Sequence to Sequence Learning with Neural Networks. *arXiv* **2014**, arXiv:1409.3215.
31. Altché, F.; de La Fortelle, A. An LSTM network for highway trajectory prediction. In Proceedings of the 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), Yokohama, Japan, 16–19 October 2017; pp. 353–359.
32. Messaoud, K.; Deo, N.; Trivedi, M.M.; Nashashibi, F. Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation. In Proceedings of the 2021 IEEE Intelligent Vehicles Symposium (IV), Nagoya, Japan, 11–17 July 2021; pp. 165–170.
33. Deo, N.; Wolff, E.; Beijbom, O. Multimodal trajectory prediction conditioned on lane-graph traversals. In Proceedings of the Conference on Robot Learning, PMLR, Auckland, New Zealand, 14–18 December 2022; pp. 203–212.
34. Schäfer, M.; Zhao, K.; Kummert, A. Caspnet++: Joint multi-agent motion prediction. In Proceedings of the 2024 IEEE Intelligent Vehicles Symposium (IV), Jeju Island, Republic of Korea, 2–5 June 2024; pp. 1294–1301.
35. Yu, D.; Lee, H.; Kim, T.; Hwang, S.H. Vehicle trajectory prediction with lane stream attention-based LSTMs and road geometry linearization. *Sensors* **2021**, *21*, 8152. [CrossRef]
36. Yoon, Y.; Kim, T.; Lee, H.; Park, J. Road-aware trajectory prediction for autonomous driving on highways. *Sensors* **2020**, *20*, 4703. [CrossRef]
37. Sadeghian, A.; Kosaraju, V.; Sadeghian, A.; Hirose, N.; Rezatofighi, H.; Savarese, S. Sophie: An attentive gan for predicting paths compliant to social and Physical Constraints. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2019; pp. 1349–1358.
38. Li, J.; Ma, H.; Tomizuka, M. Conditional generative neural system for probabilistic trajectory prediction. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), The Venetian Macao, Macau, 3–8 November 2019; pp. 6150–6156.

39. Yang, B.; He, C.; Wang, P.; Chan, C.Y.; Liu, X.; Chen, Y. TPPO: A Novel Trajectory Predictor With Pseudo Oracle. *IEEE Trans. Syst. Man Cybern. Syst.* **2024**, *54*, 2846–2859. [CrossRef]
40. Li, X.; Ying, X.; Chuah, M.C. Grip: Graph-based interaction-aware trajectory prediction. In Proceedings of the 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, 27–30 October 2019; pp. 3960–3966.
41. Jeon, H.; Choi, J.; Kum, D. SCALE-Net: Scalable vehicle trajectory prediction network under random number of interacting vehicles via edge-enhanced graph convolutional neural network. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; pp. 2095–2102.
42. Mohamed, A.; Qian, K.; Elhoseiny, M.; Claudel, C. Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 14424–14432.
43. Cui, H.; Nguyen, T.; Chou, F.C.; Lin, T.H.; Schneider, J.; Bradley, D.; Djuric, N. Deep kinematic models for kinematically feasible vehicle trajectory predictions. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 10563–10569.
44. Ścibior, A.; Lioutas, V.; Reda, D.; Bateni, P.; Wood, F. Imagining the road ahead: Multi-agent trajectory prediction via differentiable simulation. In Proceedings of the 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 19–22 September 2021; pp. 720–725.
45. Zernetsch, S.; Kohnen, S.; Goldhammer, M.; Doll, K.; Sick, B. Trajectory prediction of cyclists using a physical model and an artificial neural network. In Proceedings of the 2016 IEEE Intelligent Vehicles Symposium (IV), Gotenburg, Sweden, 19–22 June 2016; pp. 833–838.
46. Li, J.; Shi, H.; Guo, Y.; Han, G.; Yu, R.; Wang, X. Tragcan: Trajectory prediction of heterogeneous traffic agents in iov systems. *IEEE Internet Things J.* **2022**, *10*, 7100–7113. [CrossRef]
47. Messaoud, K.; Yahiaoui, I.; Verroust-Blondet, A.; Nashashibi, F. Attention based vehicle trajectory prediction. *IEEE Trans. Intell. Veh.* **2020**, *6*, 175–185. [CrossRef]
48. Hochreiter, S. Long Short-term Memory. In *Neural Computation*; MIT-Press: Cambridge, MA, USA, 1997.
49. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2013; Volume 26.
50. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; Chapter 6.
51. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation applied to handwritten zip code recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]
52. Bahdanau, D. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
53. Zhao, S.; Song, J.; Ermon, S. Infovae: Balancing learning and inference in variational autoencoders. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; Volume 33, pp. 5885–5892.
54. Oord, A.v.d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *arXiv* **2018**, arXiv:1807.03748.
55. Sohn, K.; Lee, H.; Yan, X. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2015; Volume 28.
56. Cho, K. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
57. Kalman, R.E. A new approach to linear filtering and prediction problems. *J. Basic Eng.* **1960**, *82*, 35–45. [CrossRef]
58. LaValle, S.M. *Planning Algorithms*; Cambridge University Press: Cambridge, UK, 2006.
59. Caesar, H.; Bankiti, V.; Lang, A.H.; Vora, S.; Liong, V.E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; Beijbom, O. nuScenes: A multimodal dataset for autonomous driving. *arXiv* **2019**, arXiv:1903.11027.
60. Cui, H.; Radosavljevic, V.; Chou, F.C.; Lin, T.H.; Nguyen, T.; Huang, T.K.; Schneider, J.; Djuric, N. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 2090–2096.
61. Chai, Y.; Sapp, B.; Bansal, M.; Anguelov, D. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv* **2019**, arXiv:1910.05449.
62. 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 14074–14083.