BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES
DOI: 10.24425/bpasts.2024.150811

HTTNet: Hybrid Transformer-based approaches for Trajectory prediction

Xianlei GE[1,3], Xiaobo SHEN[1,4]*, Xuanxin ZHOU[1], and Xiaoyan LI[2,3]

[1] School of Electronic Engineering, Huainan Normal University, China
[2] School of Computer, Huainan Normal University, China
[3] College of Computing and Information Technologies, National University, Philippines
[4] College of Industrial Education, Technological University of the Philippines, Philippines

Abstract. Forecasting future trajectories of intelligent agents presents a formidable challenge, necessitating the analysis of intricate scenarios and uncertainties arising from agent interactions. Consequently, it is judicious to contemplate the establishment of inter-agent relationships and the assimilation of contextual semantic information. In this manuscript, we introduce HTTNet, a comprehensive framework that spans three dimensions of information modeling: (1) the temporal dimension, where HTTNet employs a Time Encoder to articulate time sequences, comprehending the influences of past and future trajectories; (2) the social dimension, where the Trajectory Encoder facilitates the input of trajectories from multiple agents, thereby streamlining the modeling of interaction information among intelligent agents; (3) the contextual dimension, where the TF-Map Encoder integrates semantic scene input, amplifying HTTNet's cognitive grasp of scene information. Furthermore, HTTNet integrates a hybrid modeling paradigm featuring CNN and Transformer, transmuting map scenes into feature information for the Transformer. Qualitative and quantitative analyses on the nuScenes and interaction datasets highlight the exceptional performance of HTTNet, achieving 1.03 minADE10 and a 0.31 Miss Rate on nuScenes, underscoring its effectiveness in multi-agent trajectory prediction in complex scenarios.

Key words: trajectory prediction; transformer; convolutional neural network; multimodal data

1. INTRODUCTION

With the socio-economic development, automobiles have assumed a pivotal role in the daily lives of individuals, resulting in a rapid surge in per capita car ownership. According to a survey conducted by the World Health Organization (WHO), traffic accidents claim a life every 23 seconds globally, leading to an annual toll of 1.3 million fatalities, predominantly caused by vehicle collisions. The rapid advancement of cutting-edge technologies, such as artificial intelligence, has catapulted autonomous driving into the forefront of research. With its immense potential, autonomous driving technology finds extensive applications in urban transportation and intelligent logistics, empowering it to support human drivers, optimize travel routes, and thereby revolutionize traffic safety and efficiency.

Trajectory prediction stands as a critical linchpin in the paradigm of autonomous driving technology, spanning a spectrum of technical facets such as perception, prediction, and decision-making [1], [2], [3]. In the seamless operation of mobile intelligent agents like autonomous vehicles and service robots, trajectory prediction assumes a pivotal role. The essence of trajectory prediction lies in the inference of the prospective motion trajectory of an intelligent agent based on its historical motion states. Conventional methodologies predominantly exploit kinematic features, employing mathematical modeling methodologies [4] such as Kalman filtering [5] and Gaussian processes [6] to prognosticate the future trajectories of entities, including pedestrians. While these methodologies adhere to traditional approaches and demonstrate commendable temporal efficiency, they remain vulnerable to data fluctuations. In recent years, vanguard researchers have delved into the utilization of Multilayer Perceptrons (MLP) [7], [8], [9] for trajectory information modeling. However, MLP is encumbered by the terminal coordinates in trajectory sequences, resulting in a diminished reliance on long-distance information. The advent of and Long Short-Term Memory (LSTM) [10] has ushered in advancements in capturing long-distance information. For instance, Alahi et al. [11] employed LSTM for the temporal modeling of trajectory information, intensifying the network's reliance on initial coordinates. Nonetheless, LSTM face limitations due to the dimensionality of their units, aggregating solely partial long-distance information and

falling short of establishing inter-positional correlations. The introduction of Transformer [12] proffers a remedy for associating long-distance information, as self-attention mechanisms can foster relationships between each node, thereby amplifying the network's prowess in extracting information for every coordinate. However, Transformers bear a substantial computational burden. For example, AgentFormer [13] implemented a complete Transformer architecture, yielding commendable predictive outcomes but rendering it unsuitable for deployment on resource-constrained devices. Moreover, AgentFormer singularly incorporates path input and lacks the capability to establish correlations between map environments.

To address the aforementioned issues, we employ Transformer to establish relationships between multiple positional coordinates, streamlining the internal structure of Transformer layers to reduce computational overhead. Additionally, to forge connections between map and trajectory information, we leverage depth-wise separable convolutions [8] to transform image data into feature maps, which are then input into the network. The novelty of our work lies in the design of a multi-layered Transformer layer structure capable of concurrently accommodating trajectory and image information. Our primary contributions are outlined as follows:

- We propose a novel trajectory prediction network called HTTNet (Hybrid Transformer Trajectory Network), capable of assimilating information from multiple intelligent agents and map data, thereby modeling trajectory scenes.
- We introduce a TF-Map Encoder that comprehends map information and transforms it into Transformer feature maps, enhancing HTTNet's map comprehension capabilities.
- We enhance a single deep Transformer layer into multiple shallow layers, establishing a Multi-Transformer, enabling simultaneous processing of trajectory and image information.

In addition, we conducted ablation experiments, comparative experiments, and visual experiments on the nuScenes and Interaction datasets, achieving outstanding performance. The organization of the following chapters is as follows: Section 2 reviews related work, Section 3 sequentially introduces HTTNet, Encoder, Multi-Transformer, and Decoder. Experimental results and discussions are presented in Section 4 and 5.

2. RELATED WORK

A. Trajectory prediction based on time sequence.

Time series modeling is a fundamental technique that leverages the input sequence's order as a feature, proving vital for various data modalities, such as videos and audios. The trajectory prediction domain has witnessed remarkable advancements facilitated by techniques like RNN, LSTM, and GRU. In this context, trajectory data is considered sequential, leading to the widespread adoption of Recurrent Neural Network (RNN) for modeling intelligent agent trajectories. Nevertheless, RNN faces challenges like the exploding gradient problem. To overcome these obstacles, Hochreiter et al. [14] proposed the Long Short-Term Memory (LSTM) model, which effectively mitigates the exploding gradient issue. Among the recent trajectory prediction endeavors, Alahi et al. [11] introduced the Social-LSTM model, which employs LSTM to model trajectory time series data and utilizes pooling mechanisms to capture interactions among intelligent agents. However, the simplicity of Social-LSTM's interaction model has led to comparatively lower prediction accuracy. This has spurred growing enthusiasm in the trajectory prediction community for exploring time series modeling approaches. Abebe et al. [15] proposes a hybrid ARIMA-LSTM model using AIS data to accurately forecast ship trajectories in maritime transportation, enabling effective collision avoidance decision-making. [16] et al. made a significant stride with the development of the Long Short-Term Memory with Lagged Information (LAG-LSTM) model. By integrating traditional physically-driven high-speed train models with interpretable deep learning models, the LAG-LSTM effectively predicts high-speed train trajectories, outperforming other deep learning counterparts in terms of prediction accuracy. Zhang et al. [17]proposed an intelligent framework that fuses Principal Component Analysis (PCA) and Gated Recurrent Unit (GRU) in a hybrid model to predict real-time trajectory deviations during Earth Pressure Balance (EPB) tunneling. Xue et al. [18] introduced PoPPL, an innovative algorithm that classifies pedestrian trajectories into different route classes (RCs) and utilizes a bidirectional LSTM classification network to predict target regions and generate corresponding trajectories. The trajectory prediction field continues to thrive, with researchers exploring new state-of-the-art approaches and novel techniques to further advance this critical domain.

The introduction of these methods has improved the performance of neural networks in the field of trajectory prediction, contributing to the advancement of trajectory prediction. However, constrained by the size of LSTM and GRU units, the network exhibits limited capability in exploring interaction information among multiple intelligent agents. Additionally, none of these approaches have attempted to leverage maps or images to enrich the network with scene information, potentially leading to the neglect of map details. Recent works such as VectorNet [19] and LaneGCN [20] directly encode road and boundary information from HD maps into nodes of a graph, utilizing graph convolution to extract features from map information. However, map encoding cannot consider all scene information and may lose some detailed information. Therefore, designing an effective scene information extraction module remains an unresolved challenge.

B. Trajectory prediction based on Transformer.

In recent years, Transformers [12] have made remarkable strides in the domain of natural language processing, attracting increasing attention for their potential in trajectory prediction. In 2019, Zhu et al. [21] pioneered the introduction of Starnet, a Transformer-based spatiotemporal graph convolution

trajectory prediction model. Starnet effectively captures pedestrians' spatiotemporal graphs through attention mechanisms, yielding promising prediction outcomes. Zhang et al. [22] introduced the mmTransformer model, leveraging Transformers to encode and fuse vehicle and environmental information, thereby generating multimodal trajectories using path and classification loss functions. However, this approach overlooks crucial interaction factors among agents. To address temporal information loss and enhance the diversity of generated trajectories, Yuan et al. [13] investigated the impact of temporal and social dimensions on the future motion of intelligent agents, proposing the AgentFormer model. Notably, they improved the attention mechanism to focus on connections between the same agent and others, allowing for probabilistic modeling of pedestrians' potential intentions. Li et al. [23] introduced a multi-scale graph-based spatial transformer and "Memory Replay" memory graph trajectory smoothing algorithm. Their approach enables the prediction of multiple paths for historical trajectories, with an emphasis on spatial information and trajectory smoothness. Additionally, they introduced the novel evaluation metric "Percentage of Trajectory Usage" to assess diversity in multiple future predictions, achieving superior performance in multi-future prediction tasks. Jia et al. [24] presented the Heterogeneous Driving Graph Transformer (HDGT) model, which models driving scenarios as heterogeneous graphs with distinct node and edge types. By employing the Transformer structure, they successfully encoded and predicted rich and diverse semantic relationships, leading to state-of-the-art performance in trajectory prediction tasks.

While the trajectory prediction exploration within the Transformer framework remains somewhat limited, the aforementioned methods are exclusively built upon the foundational Transformer for trajectory prediction, incurring substantial temporal expenses. Additionally, these approaches have yet to delve into the amalgamation of CNN with Transformer to facilitate feature extraction from images. Acknowledging the constraints of prior trajectory prediction methodologies, we introduce HTTNet. This model adeptly accommodates both map scenes and interaction dynamics among multiple intelligent agents, presenting an innovative foray into the application of Transformer in the realm of trajectory prediction.

3. METHODOLOGY

We have formulated the prediction of future trajectories for $N(N \in Z)$ intelligent agents based on their known past trajectory distributions. Specifically, for an individual intelligent agent, we denote its state at time t as $X_t = \{X_1^t, X_2^t, X_3^t, \dots, X_N^t\}$, where $t > 0$. For an agent at a given moment X_n^t, we denote its state at time t as S_t, where S_t = {position x, position y, velocity x, velocity y, heading angle, length, width}. Similarly, the collective state of N agents within a time interval $T(T > 0)$ is represented as $Y_n = \{y_1, y_2, y_3, \dots, y_N\}$, where y_n indicates the states of the N agents during the time span T. To enable the learning model to understand past and future trajectories, we divide the trajectory over time period T into $T_1(0 \sim t')$ and $T_2(t' \sim T)$, where T_1 represents the past trajectories of the agents, and T_2 represents the future trajectories of the agents. The learning model is furnished with the past trajectories of all intelligent agents Y_n^1 within time interval T_1 and tasked with forecasting the future trajectories of all intelligent agents Y_n^2 within time interval T_2. Additionally, the learning model can obtain a semantic scene map images M, where $M \in C \times H \times W$, with C representing the number of semantic categories, H representing the image height, and W representing the image width. In the following text, we sequentially introduce the overall structure of HTTNet (Hybrid Transformer Trajectory Network.), the Trajectory Encoder, TF-Map Encoder, Multi-Transformer, and the decoder.

A. Hybrid Transformer Trajectory Network.

Our HTTNet (Hybrid Trajectory Transformer Network) is built upon the Transformer [12], a network model initially designed for natural language text processing. However, conventional Transformers are primarily tailored for handling sequential textual data and language understanding tasks. In our context, we face the challenge of modeling past trajectories, future trajectories, and the contextual information of surrounding map scenes. To address these specific requirements and enhance the modeling capabilities, we introduce two pivotal enhancements to the standard Transformer architecture:

(1) Integrated CNN and positional encoding as scene features.

(2) Reduced the number of layers and nodes within the Transformer, opting for a multi-layer cross-attention structure to facilitate the incorporation of surrounding scene image features.

As depicted in Fig. 1, HTTNet takes three distinct inputs:

(1) The past trajectories of the agents $Y = \{y_1, y_2, \dots, y_{n-1}, y_n\}$, where Y_n contains all trajectory information S_t at time t, with S_t = {position x, position y, velocity x, velocity y, heading angle, length, width};

(2) The Future trajectories of the agents $Y' = \{y_1', y_2', \dots, y_{n-1}', y_n'\}$.

(3) Semantic map image M, with $M \in C \times H \times W$.

It is worth noting that the future trajectories Y' are only input during the training process and not during testing, for guiding the model's learning process. For the past trajectories Y and future trajectories Y', we encode them for input into the Multi-Transformer for feature extraction. The traditional Transformer used a Position Embedding module for positional encoding of sequential data. However, in traffic trajectory data, each agent does not provide sequential order information, and using traditional encoding methods may lead to loss of temporal sequences. Therefore, we design a Trajectory Encoder for trajectory encoding, which includes FPN Embedding and Time Encoder, ensuring the stability of the temporal sequences of multiple agents. The Trajectory Encoder outputs features of both past and future trajectories. For the map image M, considering the time cost of the Transformer, we designed the TF-Map Encoder to extract

image features, which utilizes a lightweight CNN module to output map features. For the trajectory and map features, we use the Multi-Transformer for feature extraction. Specifically, we first input the past trajectory features into the first Cross Attention, which duplicates the features three times, named Q, K and V respectively. Then, Cross Attention uses Q to perform dot products with K and V respectively, outputting fused features to query the internal associations of the trajectory features. For the map features and future trajectory features, we first duplicate the features twice, named K and V respectively. Unlike the first Cross Attention, we use the fused features from the previous step as Q, which operates with K and V respectively, in the hope of querying the associations between the map and future trajectories from the past trajectories. The third Cross Attention outputs a feature matrix M_p that includes the agents' future trajectory features. More details on the Multi-Transformer will be introduced in Section 3.D. Finally, the Decoder module adjusts the features M_p and uses a multi-layer perceptron to transform the features into the output prediction sequence p, $p = \{p_1, p_2, \dots, p(n-1), p_n\}$.

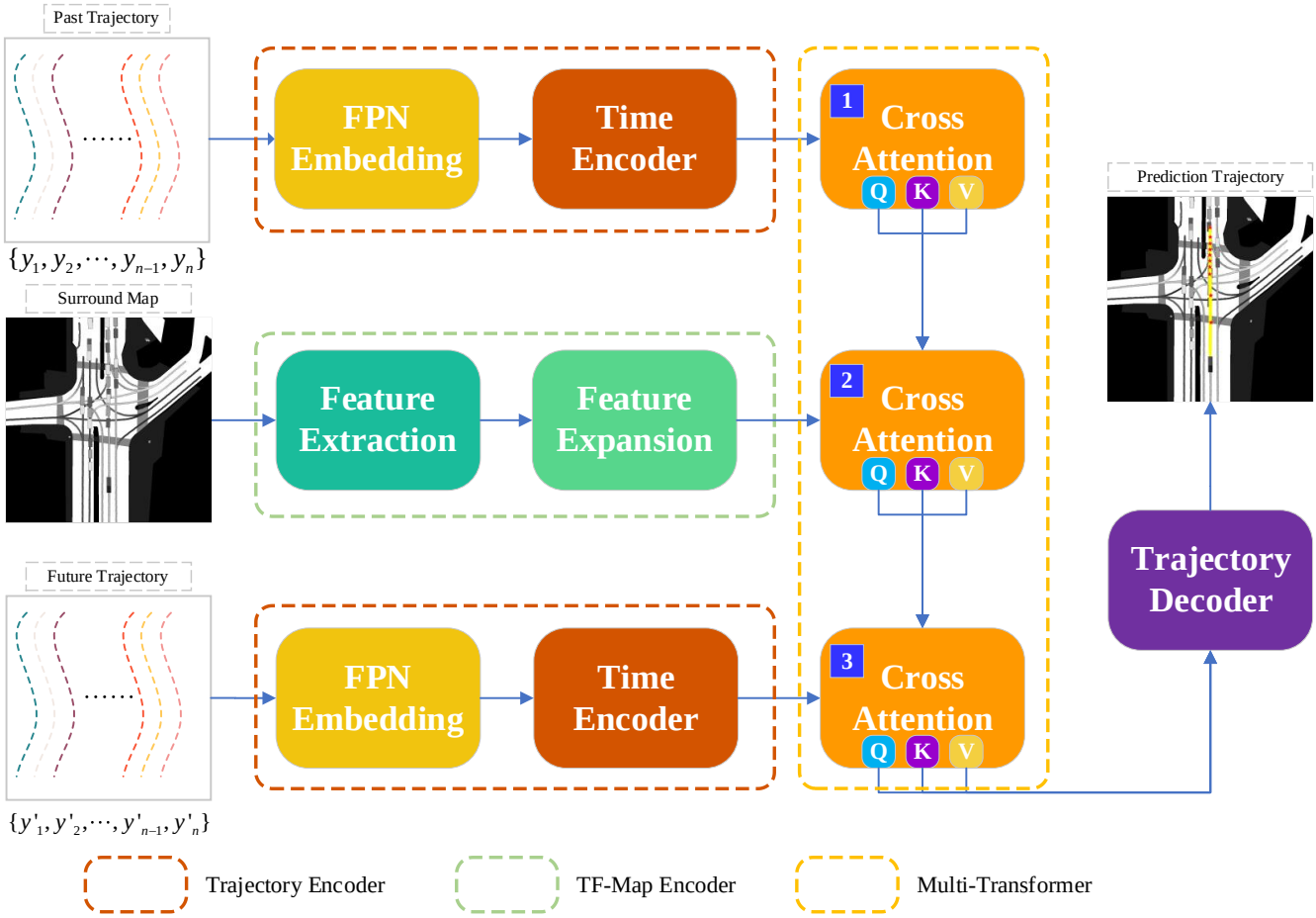

Fig.1. Overall structure of Hybrid Transformer Trajectory neural network

B. Trajectory Encoder.

To incorporate past trajectory information from time interval T into the Multi-Transformer, the trajectory data needs to be unfolded, allowing the Multi-Transformer to learn future trajectory transformation strategies. Firstly, the trajectory information from time interval T is fed into the Feature Pyramid Network (FPN), facilitating the interaction of information from different vehicles and time sequence. Moreover, inspired by the position Embedding in Transformers, we utilize a Time Encoder to treat temporal sequences as positional encoding information, which is then integrated into the trajectory information. The specific trajectory data handling operations are depicted in Fig. 2.

Specifically, regarding the input of past trajectory information for multiple intelligent agents, denoted as $Y = \{y_1, y_2, \dots, y_{n-1}, y_n\}$, where $Y \in T \times A \times D$, T represents the length of the past time interval (in frames), A represents the number of intelligent agents, and D represents the data information contained in each intelligent agent (position x, position y, etc.), a series of operations are performed. We sequentially pass Y through three Linear layers, each producing outputs with different dimensions. The first Linear layer unfolds the features, enabling a more accessible representation. The second Linear layer enhances feature perception and interaction within the hidden layer, fostering richer interactions among agents. Finally, the third Linear layer restores the original dimension D of the features. Notably, for the output of the third Linear layer, we employ zero values to expand the feature dimension D, facilitating subsequent interactions and computations. The specific

operations of FDN Embedding are mathematically expressed in Eq. (1):

$$Y' = Embiding\left(\delta^{(3)}(Y, l)\right) \quad (1)$$

where Y represents the past trajectory information, $Y \in T \times A \times D$. δ denotes the composite operation involving both $linear$ and $ReLU$ activation functions, with the second parameter specifying the output dimension of the linear layer. The $Embiding$ indicates the use of zero values to pad and extend the feature dimension D.

To incorporate time sequence information into HTTNet, we adapt the Time Encoder to encode the trajectory information. We generate a time sequence index, T_{seq}, with values ranging from 0 to 512. The length of T_{seq} is then adapt to match the time dimension. Next, we alternate between odd and even sequences of T_{seq}, dividing them by the time encoding information. The resulting values are then added to Y'. This process effectively encodes the temporal aspect of the trajectory data, enabling HTTNet to consider the temporal relationships and dynamics while predicting future trajectories. The specific operations for time encoding are in Eq. (2):

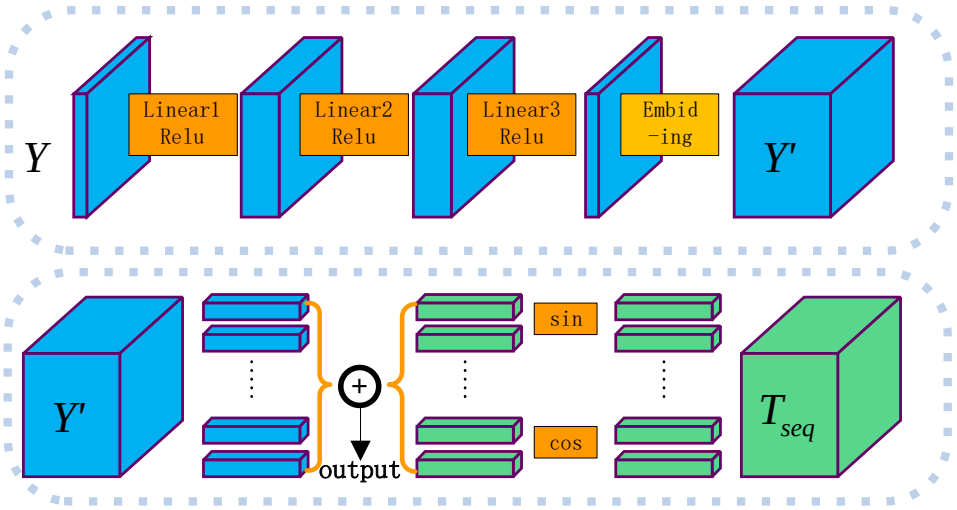

Fig.2. Trajectory Encoder structure, from top to bottom, consists of FPN and Time Encoder

$$\begin{cases} TE(T_{\sigma\varepsilon\theta}, 2i) = sin\left(\dfrac{T_{\sigma\varepsilon\theta}}{10000^{2i/d_{\nu o\delta\varepsilon\lambda}}}\right) \\ TE(T_{\sigma\varepsilon\theta}, 2i+1) = cos\left(\dfrac{T_{\sigma\varepsilon\theta}}{10000^{2i/d_{\nu o\delta\varepsilon\lambda}}}\right) \end{cases} \quad (2)$$

where TE represents the time encoder, T_{seq} represents the time sequence index, i denotes the dimension index for time encoder, and d_{model} represents the hidden layer dimension of the Transformer model. The result of TE is a d_{model} dimensional vector. Ultimately, the result of TE is added to Y' for each dimension D, yielding the past trajectory features P and the future trajectory features F.

C. TF-Map Encoder.

To input the map information into the Multi-Transformer, we use the TF-Map Encoder to encode the semantic map information. Past methods [19], [25] mainly utilized MLP (Multilayer Perceptron) and CNN (Convolutional Neural Network) for feature extraction from feature maps. However, the features from a CNN contain dimensions of height and width, which cannot be directly input into a Transformer. Therefore, we use Inverted Blocks[26] to sequentially downsample the features, reducing the dimensionality of the features. The operations of the TF-Map Encoder are shown in Figure 3.

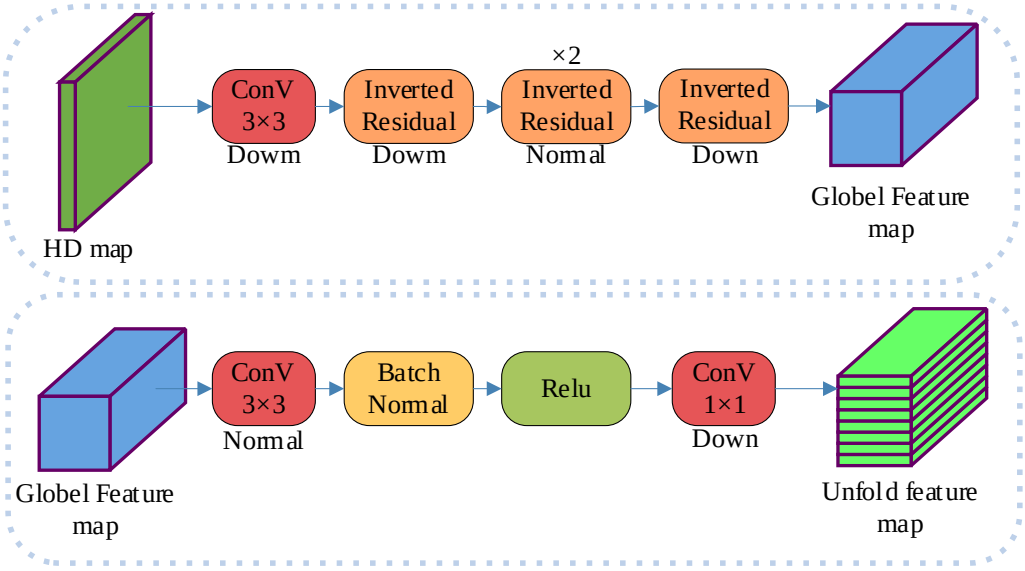

Fig.3. TF-Map encoder structure, consisting of feature extraction and feature expansion

Regarding the input feature map HD map, denoted as $M \in C \times H \times W$, where C, H, and W represent the HD map's channel number, height, and width. We conducte an adjustment through a 3×3 convolution operation. Subsequently, four Inverted Blocks are sequentially applied to perform downsampling and feature extraction on the feature map, resulting in a comprehensive Global Feature map containing all information in the semantic map. Following this, a series of operations, including 3×3 convolutions, normalization, and $ReLU$ activation, are employed to prepare the feature map for its final dimension reduction. Finally, mean pooling is applied to squeeze the $H \times W$ dimensions of the feature map, resulting in the map feature M. The calculation process of the TF-Map Encoder can be described by Eq. (3) – Eq. (5).

$$x_1 = InRes_{3\times3}^{down}\left(ConV_{3\times3}^{down}(x)\right) \tag{3}$$

$$x_2 = InRes_{3\times3}^{down}\left(InRes_{3\times3}^{2(normal)}(x_1)\right) \tag{4}$$

$$x_3 = ConV_{1\times1}^{down}\left(Relu\left(BN\left(ConV_{3\times3}^{normal}(x_2)\right)\right)\right) \tag{5}$$

where ε^{down} and ε^{normal} represents the downsampling and feature extraction operation carried out during this convolution step, ε^{i} indicates the number of ε operation. and $\varepsilon_{i\times j}$ signifies the size of the convolutional kernel. The $InRes$ represents $InvertedResidual$, consisting of two convolutional layers; details can refer to [25]. BN denotes batch normalization, and $Relu$ represents the activation function.

D. Multi-Transformer

To elucidate the correlation between the map and future trajectories from past trajectory information, we employ a Multi-Transformer for feature fusion and extraction. The efficacy of Transformers in the realm of Natural Language Processing is well-documented, chiefly attributed to their proficient processing of unimodal features via cross-attention mechanisms. However, the Multi-Transformer needs to simultaneously deal with three types of features (past trajectories, map features, and future trajectories). Therefore, we first use cross-attention to extract features from past trajectories, and then use the past trajectories to query the relationship between the map and future trajectories.

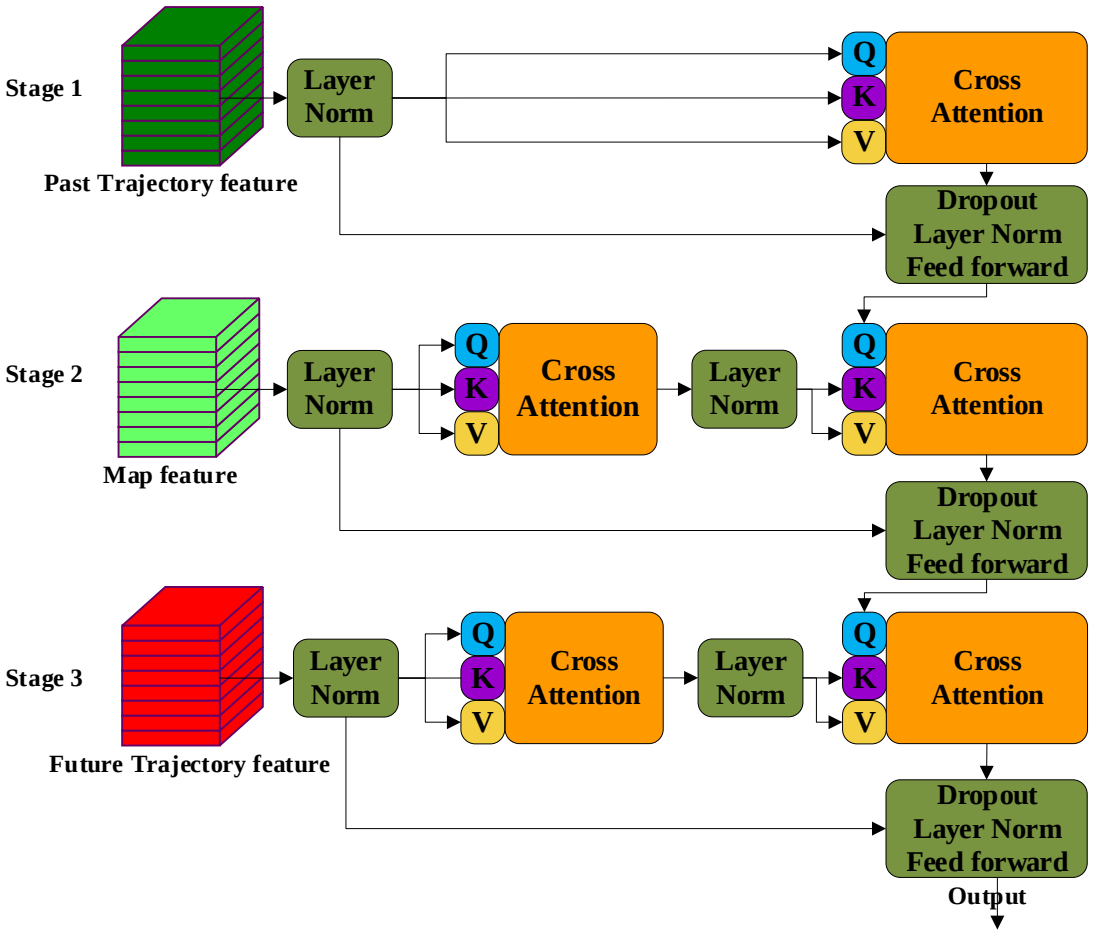

Fig.4. Multi-Transformer structure, comprising three feature extraction and fusion modules

Specifically, for the input features $P \in B \times T \times D$, $M \in B \times T \times D$, and $F \in B \times T \times D$, where P, M, and F represent past trajectory features, map features and future trajectory features; B, T and D represent the features' batch size, time length and dimension. Firstly, we normalize P using Layer Normalization and then separately input it into the Q, K, and V of the Cross-Attention, where Q, K, and V represent the query, key, and value, respectively. For the result of the cross-attention on the past trajectories, we sequentially apply Dropout, Layer Normalization, and a Feed Forward to produce the first-stage features. Similarly, we also apply Layer Normalization and cross-attention to the map feature M. Then, we normalize the result for M using Layer Normalization and input it as K and V into the cross-attention, with Q coming from the first-stage features. The cross-attention utilizes Q to query the associations between K and V, yielding the second-stage features. In a similar fashion, we perform the same operations on the future trajectory feature F, querying the association between past and future trajectories to produce the third-stage features. The calculation process of the Multi-Transformer can be described by Eq. (6) – Eq. (16):

$$x_1 = LN(P) \tag{6}$$

$$x_2 = CA(x_1, x_1, x_1) \tag{7}$$

$$x_3 = Fd\big(LayerNorm\big(dropout(x_2)\big), x_1\big) \tag{8}$$

$$y_1 = LayerNorm(M) \tag{9}$$

$$y_2 = LayerNorm\big(CA(y_1, y_1, y_1)\big) \tag{10}$$

$$y_3 = CA(x_3, y_2, y_2) \tag{11}$$

$$y_4 = Fd\big(LayerNorm\big(dropout(y_3)\big), y_1\big) \tag{12}$$

$$z_1 = LayerNorm(F) \tag{13}$$

$$z_2 = LayerNorm\big(CA(z_1, z_1, z_1)\big) \tag{14}$$

$$z_3 = CA(y_4, z_2, z_2) \tag{15}$$

$$z_4 = Fd\big(LayerNorm\big(dropout(z_3)\big), z_1\big) \tag{16}$$

where CA represents $Cross\ Attention$, consisting of three parameters: Q, K, and V, which respectively correspond to query, key, and value. The Fd represents $FeedForward$, involving a series of operations, followed by $linear$, $ReLU$, $dropout$, $linear$, $Residual\ Connections$, and $Layer\ Norm$. The second parameter of $FeedForward$ is indicative of the variable added during $Residual\ Connection$. The final output result is represented as z_4. The details about the functioning of $Cross\ Attention$ can refer to [12].

E. Trajectory Decoder.

To output the final predicted trajectory, we use Trajectory Decoder to decode the results of Multi Transformer. Specifically, for the information z_4, which is the output of the Multi-Transformer and has dimensions $z_4 \in B \times T \times D$, we employ an MLP to perform feature extraction and generate the final output. The MLP consists of a hidden layer and an output

layer. The Trajectory Decoder's specific operations can be described by Eq. (17):

$$Out = linear(relu(linear(z_4), d_1), d_2) \quad (17)$$

where z_4 is obtained from the Transformer Layer, and $d\_1 = D/2$. The d_2 represents all the trajectory information predicted for multiple intelligent agents. It is noteworthy that by adjusting the final number of channels, the model *can* predict multiple sets of trajectory information.

4. EXPERIMENT AND RESULTS

A. Dataset.

We conduct a comprehensive evaluation and ablation experiments on HTTNet using publicly available datasets, which comprise:

- Interaction dataset [26]: This dataset is tailored for behavior prediction in interactive driving scenarios, encompassing intricate urban traffic information, such as complex intersections and roundabouts, with extensive interactions among vehicles and pedestrians. Notably, the dataset offers comprehensive high-definition semantic maps, which served as the direct input source for HTTNet. Our evaluation follow the Interaction benchmark, utilizing 10 frames for training and 20 frames for prediction.
- nuScenes dataset [27]: As one of the most recent large-scale driving datasets, it boasts 1000 diverse driving scenes, encompassing a total of 28130 data samples, 3019 validation samples, and 6008 test samples. Equipped with various sensors like LIDAR, 360° cameras, and others, this dataset provides rich and varied information. Additionally, high-definition semantic maps were provided for training purposes. Our experimentation follow the nuScenes challenge benchmark, utilizing 4 frames for training and 12 frames for prediction.

B. Training details.

For all datasets, the intelligent agents' origin coordinates are established using the last frame's position coordinates from the past trajectory, denoted as (x_0, y_0). Both future and past trajectories are subsequently transformed by subtracting (x_0, y_0) from their respective coordinates. Additionally, we process other input information, such as angles, lengths, widths, and more, by computing their mean and variance. We normalize these values by mean and variance. During decoding, the normalization process is reversed using the mean, variance, and original origin coordinates. Regarding the Transformer Layer, we configure the dimensions of Query, Key, and Value to 128; the number of layers and heads are set to 2 and 4; The feedforward layers' dimension is set to 256.

We train and test HTTNet on PyTorch with an Intel W5-2465X CPU and 4 * NVIDIA GeForce RTX 4090 GPUs. The weights are initialized with Kaiming [28] initialization strategy. We use SGD optimizer with a momentum of 0.9, initial learning rate of 0.005, and weight decay of 0.0004 is employed for training. The learning rate is adjusted using the polynomial strategy, where the current learning rate is multiplied by $(1 - Curstep/TotalStep) ** 0.9$. A linear warm-up from 0.0005 to 0.05 is used for the first 1000 steps. For the loss function, Mean Squared Error (MSE) is used for computation. For the Interaction dataset, we set the batch to 16 and trained a total of 20000 steps, which took 3.7 hours.For nuScenes, we set the batch to 32 and trained a total of 40000 steps, which took 12.3 hours.

C. Evaluation metrics.

We employed well-established trajectory prediction evaluation metrics, namely the Minimum Average Displacement Error (minADE), the Minimum Final Displacement Error (minFDE), and the Miss Rate ($MissedPredictions/GroundTruth$).The specific formulas for minADE and minFDE are as follows:

- minADE: Consider N ground truth trajectories and their corresponding K predicted trajectories, each comprising T time steps. For the i-th ground truth trajectory, let $P(i,t)$ represent its true coordinates (x, y) at the t-th time step, and let $P_{pred}(i,t)$ denote the corresponding predicted trajectory. The calculation formula for minADE can be described as Eq. (18):

$$\min \text{ADE} = \frac{1}{T} \min_{i=1}^{K} \sum_{t=1}^{T} \left| P(i,t) - P_{\text{pred}}(i,t) \right| \quad (18)$$

where $\min_{i=1}^{K}$ finds the minimum error value among all K ground truth trajectories, $\sum_{t=1}^{T}$ represents the summation over T time steps for each trajectory. The $| \, |$ signifies the Euclidean distance.

- minFDE: N, K, $P(i,t)$, (x, y) and $P_{pred}(i,t)$ are defined in the same way as in minADE. The calculation formula for minFDE can be described as Eq. (19):

$$\min \text{FDE} = \frac{1}{K} \min_{i=1}^{K} \left| P(i,T) - P_{\text{pred}}(i,T) \right| \quad (19)$$

D. Ablation Study.

(1) Ablation Encoder

In this section, we present the design of ablation experiments conducted on the Encoder component of the HTTNet model. HTTNet represents a novel hybrid approach that combines the Transformer and CNN architectures to address the challenging task of multi-agent trajectory prediction. To comprehensively assess the influence of the Encoder on trajectory prediction performance, we conducted separate ablation analyses on the TF-Map Encoder and future encoder using two datasets: the Interaction dataset and nuScenes dataset. For evaluation metrics, we employ minADE and minFDE, with trajectory generation quantities set at K=5 and K=10, respectively. Table 1 displays the results of the Encoder ablation experiments, with the corresponding symbols defined as follows:

Base: Includes Trajectory Encoder
Map: Includes TF-Map Encoder
Future: Includes Future Trajectory encoder

TABLE 1

Ablation study of encoder on Interaction and nuScenes

Dataset	Base	Map	Future	minADE5	minFDE5	minADE10	minFDE10
Interaction	√			0.43	0.78	0.21	0.62
	√	√		0.41	0.79	0.25	0.62
	√		√	0.24	0.63	0.18	0.51
	√	√	√	0.17	0.48	0.15	0.46
nuScenes	√			2.03	4.78	1.44	4.31
	√	√		1.97	4.81	1.45	4.33
	√		√	1.57	3.82	1.21	3.11
	√	√	√	1.23	3.21	1.03	2.57

As shown in Table 1, HTTNet achieve superior performance consistently on both the Interaction dataset and the nuScenes dataset when all encoders were included. Comparing the scenarios of including the TF-Map Encoder versus solely employing the Trajectory Encoder, the impact of the TF-Map Encoder on HTTNet's performance is found to be relatively modest on both datasets. In fact, in the case of minFDE5 on the Interaction dataset, the addition of the TF-Map Encoder even led to a slight increase in the metric. On the other hand, contrasting the inclusion of the Future Encoder with the sole use of the Trajectory Encoder, we observe notable performance gains on both datasets when incorporating the Future Encoder. This observation indicates that the future features introduce by the Future Encoder play a significant role in guiding the model's learning process. Notably, when all Encoders were utilized, the model demonstrate the most significant performance improvements on both datasets. These findings confirm the efficacy of HTTNet in effectively leveraging past and future trajectory information from multiple intelligent agents, as well as map image information, to achieve enhanced trajectory prediction results.

(2) Ablation Transformer Layers and heads

The complexity of Multi-Transformer is governed by the number of Transformer Layers and heads. Given the diverse practical scenarios, it becomes imperative to judiciously select the appropriate number of layers and heads to strike a balance between model complexity and predictive accuracy. In this section, we conduct ablation experiments on HTTNet, varying the numbers of layers and heads, and evaluate its performance on the Interaction dataset. It is crucial to emphasize that throughout the experiments pertaining to Transformer Layers and heads, all encoders are integrated into the HTTNet architecture. The specific outcomes of these experiments are meticulously presented in Table 2.

The results from Table 2 highlight the substantial impact of varying Transformer Layers and heads on the performance of the HTTNet. When HTTNet is equipped with a single Layer and a limited number of Heads (e.g. 2), the model exhibits relatively lower complexity. Despite yielding relatively higher minADE and minFDE values, the model's predictions still achieve a certain level of accuracy. As we increase the number of Layers (≤ 2) and Heads (≤ 4), the model's complexity gradually escalates, resulting in significant reductions in minADE and minFDE, thus indicating an enhanced predictive accuracy with more Layers and Heads. However, further augmenting the number of Layers (> 2) and Heads (> 4) eventually leads the model to reach an equilibrium. Consequently, minADE and minFDE continue to decrease, but the rate of decrease becomes less pronounced compared to earlier increments. Specifically, when Layer=3 and Heads=6, no significant improvement in the model's performance is observed. On the contrary, minADE and minFDE show a slight upward trend, suggesting that an excessive number of Layers and Heads might have minimal or negligible effect on the model's performance or, in some cases, even yield adverse outcomes. Based on the insights gained from the ablation experiments, we ultimately opt for Layer=2 and Head=4 as the optimal configuration for HTTNet.

TABLE 2

Ablation study of Transformer layers and heads on Interaction

Dataset	layer	head	minADE5	minFDE5	minADE10	minFDE10
Interaction	1	2	0.62	1.11	0.93	1.57

	1	3	0.37	0.78	0.37	0.88
	2	3	0.25	0.71	0.24	0.66
	2	4	0.19	0.52	0.15	0.47
	3	4	0.18	0.48	0.14	0.44
	3	5	0.18	0.47	0.14	0.45
	3	6	0.17	0.48	0.13	0.47

5. COMPARATIVE STUDY

To further assess the trajectory prediction performance of the HTTNet model, we conducted comparative analyses with state-of-the-art models renowned for their performance in this domain. The selected models included Trajectron++ [29], P2T [30], AgentFormer [13], LaPred [31], MultiPath [32], GOHOME [33], ThOMAS [34], PGP [35], DESIRE [36], TNT [37] and ITRA [38]. We compared the results of these models with HTTNet on both the nuScenes dataset and the Interaction dataset. The comparative experiment outcomes are presented in Table 3 and Table 4, respectively.

TABLE 3

Comparative study with other models on nuScenes

Method	MADE5	MADE10	MR5	MR10
Trajectron++	1.88	1.51	0.7	0.57
P2T	1.45	1.16	0.64	0.46
AgentFormer	1.86	1.45	-	-
LaPred	1.47	1.12	0.53	0.46
MultiPath	1.44	1.14	-	-
GOHOME	1.42	1.15	0.57	0.47
ThOMAS	1.33	1.04	0.55	0.42
PGP	1.27	0.94	0.52	0.34
HTTNet	1.23	1.03	0.49	0.31

TABLE 4

Comparative study with other models on Interaction

Method	MADE6	MFDE10
DESIRE	0.32	0.88
MultiPath	0.3	0.99
TNT	0.24	0.67
GOHOME	-	0.45
ITRA	0.17	0.49
HTTNet	0.16	0.47

The comparative study results in Table 3 demonstrate the remarkable performance advantages of the HTTNet model on the nuScenes dataset. When predicting samples with 5 and 10, the HTTNet model achieved minADE values of 1.23 and 1.03, respectively, which are significantly lower than those of other models like Trajectron++, P2T, and LaPred. Additionally, we evaluated the model's Miss Rate (MR), which indicates the number of trajectories in the predictions that exceed a certain threshold distance from the ground truth trajectories. The HTTNet model also exhibited exceptional performance on MR5 and MR10, with values of 0.49 and 0.31, respectively.

In the comparative study results presented in Table 4, the HTTNet model demonstrated outstanding performance on the Interaction dataset. For predicting samples with 5 and 10, the minADE and minFDE values were 0.16 and 0.47, respectively. These values are lower than those achieved by other models on these two metrics. Considering the comprehensive comparative study results with other popular models on both the nuScenes and Interaction datasets, it becomes evident that the HTTNet model holds a substantial advantage in the field of trajectory prediction.

6. VISUALIZATION OF DIFFERENT MODULES

To validate the effectiveness of each module in HTTNet, we conducted a visual analysis on the nuScenes dataset. Specifically, we sequentially visualized three ablation cases—Base, Base+Future, and Base+Future+Map—onto the semantic map, with the generated number of trajectories selected as 10. To observe the differences in the final coordinates predicted for each trajectory, we utilized asterisks (*) for description.

By reviewing Fig. 5, we can observe that Base+Future+Map has achieved the best performance, both in terms of endpoint coordinates and the distance from the original trajectories. Comparing Base with Base+Future, we notice significant trajectory dispersion in Base, while Base+Future constrains the trajectories. This is attributed to the guiding influence of future trajectories during training. Furthermore, in the comparison of the three results, trajectories in Base+Future+Map consistently remain within the semantic roads. This is because the semantic map imposes constraints on the predicted trajectories, whereas the other two cases exhibit instances of trajectories deviating from semantic roads. In conclusion, HTTNet excels in both map boundaries and trajectory scope.

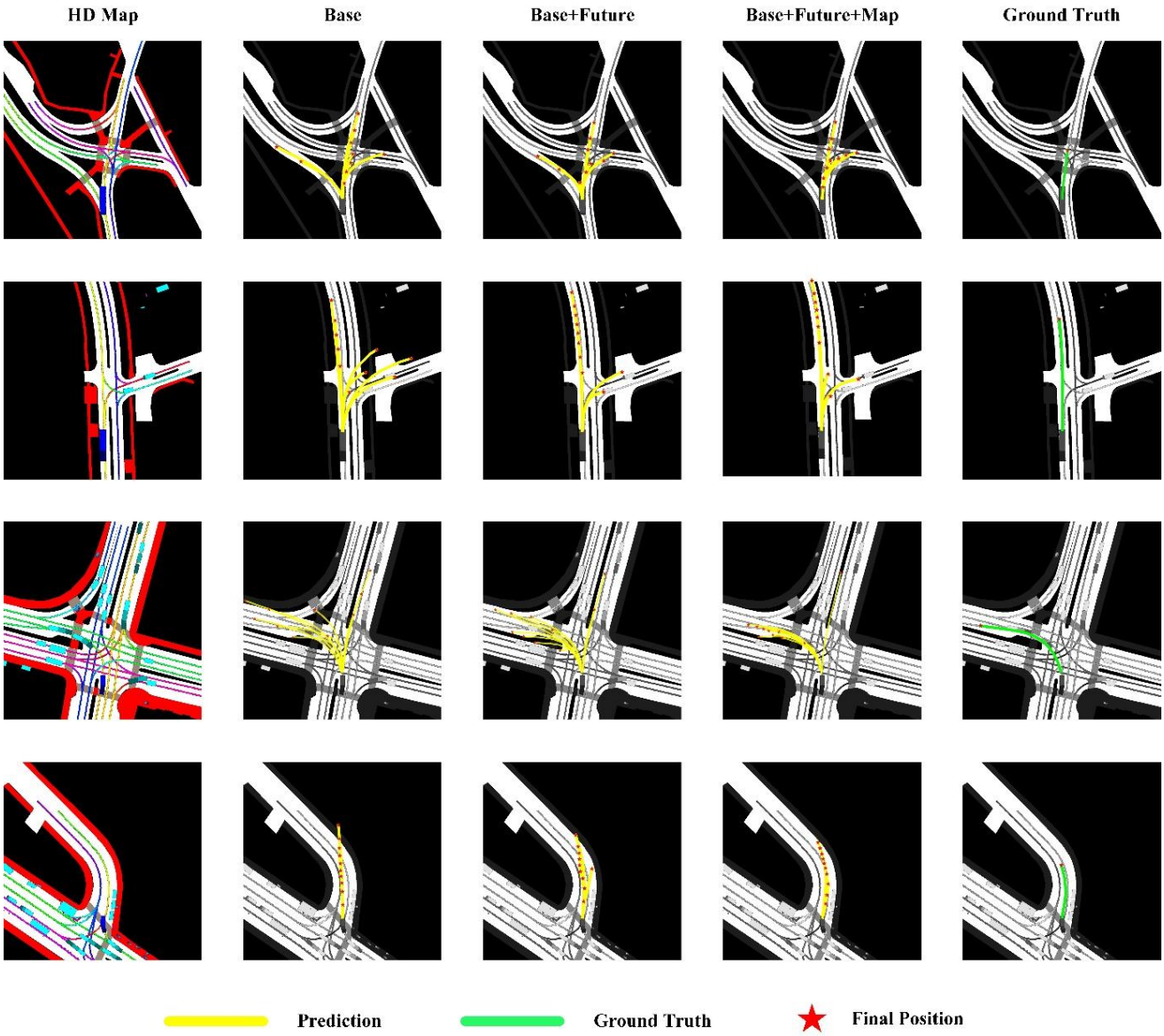

Fig.5. Visualization of different modules(sample=6)

7. CONCLUSION

In this paper, we introduce a novel trajectory prediction model named HTTNet, which is based on Transformer and CNN. HTTNet comprises three main modules: Trajectory Encoder, TF-Map Encoder, Multi-Transformer, and Trajectory Decoder. Specifically, the Trajectory Encoder extracts low-level trajectory information and encodes it, while the TF-Map Encoder encodes map information, enhancing HTTNet's understanding of road boundaries. The Multi-Transformer receives different features from the Encoders, performs extraction and fusion, and, finally, the Trajectory Decoder organizes feature information into trajectory data for output. We conducted qualitative and quantitative analyses on HTTNet, demonstrating the effectiveness of its structure. Remarkably, we achieved outstanding performance on challenging datasets, namely Interaction and nuScenes. Our work paves the way for the application of Transformer in trajectory prediction, significantly exploring the synergy of CNN and Transformer methods. The proposed HTTNet can be applied to various downstream tasks such as autonomous driving and behavioral decision-making. Our future work aims to extend predictions to multiple agents, simulating interactions between each intelligent agent.

ACKNOWLEDGEMENTS

This research is supported by the University Natural Science Foundation of Anhui Province (Grant No.2022AH051578, 2023AH051551) and Guiding Science and Technology Foundation of Huainan (Grant No.2020050).

REFERENCES

[1] H. Cui *et al.*, “Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks,” in *2019 International Conference on*

Robotics and Automation (ICRA), 2019, pp. 2090–2096. doi: 10.1109/ICRA.2019.8793868.

[2] F. Leon and M. Gavrilescu, "A review of tracking and trajectory prediction methods for autonomous driving," *Mathematics*, vol. 9, no. 6. 2021. doi: 10.3390/math9060660.

[3] Z. Sheng, Y. Xu, S. Xue, and D. Li, "Graph-Based Spatial-Temporal Convolutional Network for Vehicle Trajectory Prediction in Autonomous Driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17654–17665, Oct. 2022, doi: 10.1109/TITS.2022.3155749.

[4] Y. Zhang, W. Wang, R. Bonatti, D. Maturana, and S. Scherer, "Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories," in *Proceedings of Machine Learning Research*, ML Research Press, 2018, pp. 894–905.

[5] S. J. Qiao, N. Han, X. W. Zhu, H. P. Shu, J. L. Zheng, and C. A. Yuan, "A Dynamic Trajectory Prediction Algorithm Based on Kalman Filter," *Tien Tzu Hsueh Pao/Acta Electronica Sinica*, vol. 46, no. 2, 2018, doi: 10.3969/j.issn.0372-2112.2018.02.022.

[6] H. Rong, A. P. Teixeira, and C. Guedes Soares, "Ship trajectory uncertainty prediction based on a Gaussian Process model," *Ocean Engineering*, vol. 182, 2019, doi: 10.1016/j.oceaneng.2019.04.024.

[7] S. Becker, R. Hug, W. Hübner, and M. Arens, "An Evaluation of Trajectory Prediction Approaches and Notes on the TrajNet Benchmark." 2018.

[8] M. M. Kordmahalleh, M. G. Sefidmazgi, and A. Homaifar, "A sparse recurrent neural network for trajectory prediction of atlantic hurricanes," in *GECCO 2016 - Proceedings of the 2016 Genetic and Evolutionary Computation Conference*, 2016. doi: 10.1145/2908812.2908834.

[9] E. Lukasik *et al.*, "Recognition of handwritten Latin characters with diacritics using CNN," *Bulletin of the Polish Academy of Sciences Technical Sciences*, vol. 69, no. No. 1, pp. e136210–e136210, 2021, doi: 10.24425/bpasts.2020.136210.

[10] J. Wróbel and A. Kulawik, "Influence of modelling phase transformations with the use of LSTM network on the accuracy of computations of residual stresses for the hardening process," *Bulletin of the Polish Academy of Sciences Technical Sciences*, vol. 71, no. 4, pp. e145681–e145681, 2023, doi: 10.24425/bpasts.2023.145681.

[11] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016. doi: 10.1109/CVPR.2016.110.

[12] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[13] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "AgentFormer: Agent-Aware Transformers for Socio-Temporal Multi-Agent Forecasting," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021. doi: 10.1109/ICCV48922.2021.00967.

[14] A. Graves, "Supervised Sequence Labelling," 2012. doi: 10.1007/978-3-642-24797-2_2.

[15] M. Abebe, Y. Noh, Y. J. Kang, C. Seo, D. Kim, and J. Seo, "Ship trajectory planning for collision avoidance using hybrid ARIMA-LSTM models," *Ocean Engineering*, vol. 256, 2022, doi: 10.1016/j.oceaneng.2022.111527.

[16] J. Yin, C. Ning, and T. Tang, "Data-driven models for train control dynamics in high-speed railways: LAG-LSTM for train trajectory prediction," *Inf Sci (N Y)*, vol. 600, 2022, doi: 10.1016/j.ins.2022.04.004.

[17] N. Zhang, N. Zhang, Q. Zheng, and Y. S. Xu, "Real-time prediction of shield moving trajectory during tunnelling using GRU deep neural network," *Acta Geotech*, vol. 17, no. 4, 2022, doi: 10.1007/s11440-021-01319-1.

[18] H. Xue, D. Q. Huynh, and M. Reynolds, "PoPPL: Pedestrian Trajectory Prediction by LSTM with Automatic Route Class Clustering," *IEEE Trans Neural Netw Learn Syst*, vol. 32, no. 1, 2021, doi: 10.1109/TNNLS.2020.2975837.

[19] J. Gao *et al.*, "VectorNet: Encoding HD maps and agent dynamics from vectorized representation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020. doi: 10.1109/CVPR42600.2020.01154.

[20] M. Liang *et al.*, "Learning Lane Graph Representations for Motion Forecasting," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020. doi: 10.1007/978-3-030-58536-5_32.

[21] Y. Zhu, D. Qian, D. Ren, and H. Xia, "StarNet: Pedestrian Trajectory Prediction using Deep Neural Network in Star Topology," in *IEEE International Conference on Intelligent Robots and Systems*, 2019. doi: 10.1109/IROS40897.2019.8967811.

[22] Y. Liu, J. Zhang, L. Fang, Q. Jiang, and B. Zhou, "Multimodal Motion Prediction with Stacked Transformers," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021. doi: 10.1109/CVPR46437.2021.00749.

[23] L. Li, M. Pagnucco, and Y. Song, "Graph-based Spatial Transformer with Memory Replay for Multi-future Pedestrian Trajectory Prediction," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2022. doi: 10.1109/CVPR52688.2022.00227.

[24] X. Jia, P. Wu, L. Chen, Y. Liu, H. Li, and J. Yan, "HDGT: Heterogeneous Driving Graph Transformer for Multi-Agent Trajectory Prediction via Scene Encoding," *IEEE Trans Pattern Anal Mach Intell*, vol. 45, no. 11, 2023, doi: 10.1109/TPAMI.2023.3298301.

[25] Mark Sandler, A. Howard, M. Zhu, A. Zhmoginov, and Liang-Chieh Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks Mark," *Convolutional Neural Networks with Swift for Tensorflow*, 2019.

[26] W. Zhan *et al.*, "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps," *CoRR*, vol. abs/1910.03088, 2019, [Online]. Available: http://arxiv.org/abs/1910.03088

[27] H. Caesar *et al.*, "Nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020. doi: 10.1109/CVPR42600.2020.01164.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034. doi: 10.1109/ICCV.2015.123.

[29] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020. doi: 10.1007/978-3-030-58523-5_40.

[30] N. Deo and M. M. Trivedi, "Trajectory Forecasts in Unknown Environments Conditioned on Grid-Based Plans," *CoRR*, vol. abs/2001.00735, 2020, [Online]. Available: http://arxiv.org/abs/2001.00735

[31] B. Do Kim *et al.*, "Lapred: Lane-aware prediction of multi-modal future trajectories of dynamic agents," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021. doi: 10.1109/CVPR46437.2021.01440.

[32] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction," in *Proceedings of Machine Learning Research*, 2019.

[33] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "GOHOME: Graph-Oriented Heatmap Output for future Motion Estimation," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2022. doi: 10.1109/ICRA46639.2022.9812253.

[34] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "THOMAS: Trajectory Heatmap Output with learned Multi-Agent Sampling," *CoRR*, vol. abs/2110.06607, 2021, [Online]. Available: https://arxiv.org/abs/2110.06607

[35] N. Deo, E. Wolff, and O. Beijbom, "Multimodal Trajectory Prediction Conditioned on Lane-Graph Traversals," in *5th Annual Conference on Robot Learning* , 2021. [Online]. Available: https://openreview.net/forum?id=hu7b7MPCqiC

[36] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "DESIRE: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017. doi: 10.1109/CVPR.2017.233.

[37] H. Zhao *et al.*, "TNT: Target-driveN Trajectory Prediction." 2020.

[38] A. Scibior, V. Lioutas, D. Reda, P. Bateni, and F. Wood, "Imagining the Road Ahead: Multi-Agent Trajectory Prediction via Differentiable Simulation," in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2021. doi: 10.1109/ITSC48978.2021.9565113.