IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 25, 2021, accepted June 21, 2021, date of publication June 25, 2021, date of current version July 6, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3092515

Dual-Channel and Bidirectional Neural Network for Hypersonic Glide Vehicle Trajectory Prediction

YANGFAN XIE, XUEBIN ZHUANG, ZEPU XI, AND HONGBO CHEN
School of System Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China
Corresponding author: Xuebin Zhuang (zhuangxb@mail.sysu.edu.cn)

This work did not involve human subjects or animals in its research.

ABSTRACT Different from traditional aircraft, hypersonic glide vehicles (HGVs) possess stronger maneuverability and a higher flight speed (generally higher than 5 Mach), making trajectory prediction very complicated. Several works have been conducted in this field, which usually analyze the motion characteristics of the HGV first and then use a Kalman filter to track and predict the trajectory. In this way, the accuracy of prediction depends on how to model the control parameters of the target vehicle. The core idea of this paper mainly concerns treating the HGV trajectory prediction as a multivariate time series forecasting problem since HGV trajectories are special multivariate time series with nonperiodic temporal patterns. Moreover, capturing the hidden dependencies between time steps and different time series ensures the accuracy and robustness of predictions. Recently, recurrent neural networks have been widely used in predicting data with temporal patterns between several time steps; however, they fail in capturing nonperiodic temporal patterns. Therefore, we propose a brand-new model named the dual-channel and bidirectional neural network (DCBNN) to intelligently predict the trajectory of hypersonic vehicles in undetectable areas, especially for those with complex maneuver models. DCBNN is constructed with a nonlinear component and a linear component to collect both nonlinear and linear features from input data to improve robustness. Moreover, a dual-branch architecture is utilized in the nonlinear component to capture the complex and mixed dependencies between long-term and short-term patterns. Experiments reveal that the proposed method is effective and intelligent.

INDEX TERMS Hypersonic glide vehicle, trajectory prediction, deep learning, time series forecasting, bidirectional gated recurrent unit.

I. INTRODUCTION

Hypersonic glide vehicles (HGVs) are a kind of aerial vehicle that uses aerodynamic forces to perform a long-range glide with high speed (usually greater than 5 Mach). This high speed enables the vehicle to shorten the duration of flight, conduct stronger maneuvers, and surpass conventional aircraft in many aerospace fields. At the same time, the high flight speed also brings much difficulty in trajectory prediction. One typical kind of hypersonic glide vehicle is common aero vehicle (CAV) [1].

Since an HGV flies in near space, the characteristics of its working environment are complex because the density of the atmosphere changes dramatically with altitude, resulting in trouble in trajectory prediction. The technology of tracking and prediction for HGVs is the premise for achieving accurate control and efficient defense. Previous studies of HGV trajectory prediction focused on analyzing the motion characteristics of the target vehicle with known aerodynamic characteristic parameters and a particular maneuver model. Using a polynomial model [2], [3] to predict trajectories has been proven to be feasible, which works by modeling the vehicle motion process with an n-degree polynomial and then integrating the kinetic equation. Recently, the work in [4] studied the control quantity prediction model based on conditional random field (CRF) theory to predict the vehicle trajectory in a blind area. In fact, control parameters of target hypersonic vehicles are difficult to identify precisely, which calls for new breakthroughs in trajectory prediction methods.

The associate editor coordinating the review of this manuscript and approving it for publication was Kah Phooi Seng.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

To predict the trajectory of HGVs more precisely, we adopt the idea of multivariate time series forecasting. Multivariate time series forecasting problems have recently attracted wide attention and been applied in many areas, such as traffic systems [5], sensor networks [6], financial problems [7] and so on. The major task of multivariate time series forecasting is to capture the complex and mixed dependencies that exist not only in time steps but also between different time series, which is similar to identifying the control parameters and modeling the kinematic characteristic in trajectory prediction. To adopt multivariate time series forecasting methods to predict the trajectory of hypersonic vehicles, we need to focus attention on the dynamic change in control parameters, which means that the former dependencies captured from trajectory data are no longer applicable when control parameters change. Traditional trajectory prediction methods often assume that the control parameters will not change or will change in a typical way, resulting in the failure of capturing dynamic temporal patterns of trajectory data and limitations in application scenarios. Moreover, traditional approaches such as vector autoregression (VAR) [8] and Gaussian process (GP) [9] methods in the multivariate time series forecasting field fall short in this aspect, which reduces the accuracy of forecasting.

With the development of deep learning, neural networks have been introduced into multivariate time series forecasting tasks by researchers and have made extraordinary impacts due to their capability of capturing nonlinearity. In particular, the recurrent neural network (RNN) [10] is considered a leader in the trend of sequence modeling. However, as a result of vanishing gradients [11], traditional RNNs fail to capture very long-term dependencies. Moreover, to overcome this drawback, the long short-term memory (LSTM) [12] network has been proposed as a variant and has achieved good performance in various deep learning fields such as natural language processing (NLP) [13], speech recognition [14] and conventional aircraft trajectory prediction [15]. The special gate mechanism in recurrent units assists the LSTM model to learn very long-term patterns of time series across a large time scope and remarkably enhances the performance of prediction by alleviating the gradient vanishing problem. On the other hand, configuring the complex parameters of the gate mechanism consumes much computational resources and leads to a decrease in the prediction speed. Therefore, the gated recurrent unit (GRU) [16], an optimized version of the LSTM model, is presented to simplify the complicated architecture of LSTM cells while both accelerating the convergence speed and maintaining a forecasting accuracy similar to that of LSTM. In addition, the convolutional neural network (CNN) has been proven to be prominent in the field of computer vision [17] by extracting features at various levels from input images and can be used to classify hypersonic trajectories [18].

The main contributions of this paper are described as follows:

- For accurate and robust HGV trajectory prediction, we propose a deep learning framework named the dual-channel and bidirectional neural network (DCBNN). The DCBNN consists of two parallel components, namely, the nonlinear pattern component and linear pattern component, to enhance the robustness by both modeling the nonlinearity and linearity of the input trajectory data. The nonlinear pattern component utilizes two aligned channels, called the global temporal channel and local temporal channel. Both channels are constructed with a convolutional layer and a bidirectional gated recurrent unit (Bi-GRU) [19]. To further improve the performance, DCBNN incorporates six parallel dense layers as a linear pattern component alongside the nonlinear pattern component.
- Considering the real-time requirement of trajectory prediction, which limits the time consumption of the prediction process, we divide the training process of DCBNN into two steps, namely, pretraining and retraining. By using a large quantity of historical trajectory data to train the model, pretraining helps DCBNN learn the hidden dependencies and realize trajectory prediction. Moreover, the retraining process facilitates improving the accuracy of trajectory prediction by retraining the model with real trajectory data observed from target vehicles.
- We compare the proposed method with six other models on three different trajectory data sets that are constructed with different maneuver models. The experimental results reveal that the accuracy of DCBNN in real-time trajectory prediction is improved with the well-designed dual-channel architecture.

The remainder of this paper is organized as follows. Section II describes the details of the problem and data processing. Section III introduces our DCBNN model. Section IV shows the results of the prediction experiment and the comparison between DCBNN and six other models. Finally, this paper is concluded in Section V.

II. PRELIMINARIES

In this section, we introduce the deduced dynamic models of an HGV employed in this paper and describe three kinds of maneuver models designed to imitate different application scenarios. Finally, the process of constructing simulated data sets of the three maneuver models is discussed.

A. DYNAMIC MODELS

Taking the rotation and nonspherical perturbations of the Earth into account will introduce considerable trouble in describing the HGV motion, while they generate little effect on forecasting. Therefore, we adopt the HGV motion equations deduced by the study [20] with the assumption that the Earth is a homogeneous sphere without rotation and that the sideslip angle is zero. The motion equations of an HGV are

of the form

$$\dot{r} = V\sin\theta, \tag{1}$$
$$\dot{\lambda} = \frac{-V\cos\theta\sin\sigma}{r\cos\varphi}, \tag{2}$$
$$\dot{\varphi} = \frac{V\cos\theta\cos\sigma}{r}, \tag{3}$$
$$\dot{V} = -D - g\sin\theta, \tag{4}$$
$$\dot{\theta} = \frac{L\cos v + (V^2/r - g)\cos\theta}{V}, \tag{5}$$
$$\dot{\sigma} = \frac{L\sin v}{V\cos\theta} + \frac{V\tan\varphi\cos\theta\sin\sigma}{r}. \tag{6}$$

where $r, \lambda, \varphi, V, \theta,$ and σ represent the radial distance from the vehicle to the center of the Earth, longitude, latitude, speed, ballistic inclination and the course angle that is measured from the North in a clockwise direction, respectively. v is the bank angle that constitutes control parameters along with angle of attack α. g is the gravitational constant of acceleration. L, D indicate the lift and drag accelerations, respectively, and are defined as follows:

$$L = \frac{C_L\rho V^2 S_M}{2m}, \tag{7}$$
$$D = \frac{C_D\rho V^2 S_M}{2m}. \tag{8}$$

where S_M is the reference area in m^2, m is the vehicle mass in kg and the atmosphere density model ρ comes from the United States Standard Atmosphere (USSA). In addition, C_L and C_D are lift and drag coefficients, respectively, that are associated with the angle of attack α and Mach number Ma as shown in equations (9,10), respectively, making L, D nonlinear functions of α, Ma.

$$C_L = f_1(M_a, \alpha) \tag{9}$$
$$C_D = f_2(M_a, \alpha) \tag{10}$$

B. MANEUVER MODELS

To comprehensively test the performance of the proposed method in predicting trajectories, we design three kinds of maneuver models, namely, a longitudinal-only maneuver model, a longitudinal with lateral turning maneuver model, and a longitudinal with lateral weaving maneuver model. In this part, the HGV maneuver model is divided into the longitudinal maneuver and lateral maneuver, and all maneuver models share a common longitudinal maneuver model.

1) LONGITUDINAL MANEUVER MODELS

To meet the constraints involving the aerodynamic press, load factor, and heating rate during the course of flight, we design the angle of attack α, the control parameter of the longitudinal maneuver, as a piecewise linear function of the speed of vehicle. The function can be described as the following equation:

$$\alpha = \begin{cases} \alpha_{max}, & V \geq V_2 \\ \dfrac{V - V_1}{V_2 - V1}(\alpha_{max} - \alpha_{L/D}) + \alpha_{L/D}, & V_1 \leq V \leq V_2 \\ \alpha_{L/D}, & V \leq V_1 \end{cases} \tag{11}$$

where α is the angle of attack. α_{max} is the upper bound of the angle of attack, and $\alpha_{L/D}$ represents the angle of attack that holds the maximum lift-to-drag ratio. V_1 and V_2 are the lower and upper threshold of velocity, respectively. Setting $\alpha_{max} = 25°$, $V_1 = 3000$ m/s, and $V_2 = 5000$ m/s, $\alpha_{L/D}$ is calculated to be 11°. Fig. 1 provides an illustration to help explain the relationship between the angle of attack and velocity.

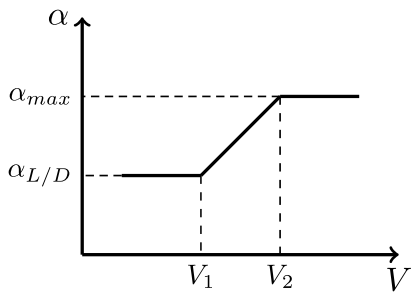


FIGURE 1. $\alpha - V$ **diagram.**

2) LATERAL MANEUVER MODELS

Three lateral maneuver models are designed to imitate various situations and exhaustively show the capability of proposed method to predict trajectories with complex maneuver models. The control parameter of the lateral maneuver is bank angle v, which is defined as piecewise linear functions of time. The first lateral maneuver model is the simplest model with its bank angle v constantly set to be zero, as shown in Fig. 2b. The equation is

$$v = v_0, \quad t \in [0, 2000]. \tag{12}$$

where t is a time step within $[0, 2000]$, and 2000 is the maximum length of the trajectory data. v_0 is set to be 0°.

The lateral turning maneuver model simulates the situation where the HGV turns in a certain direction, whose bank angle v increases to 20° at 500 s and returns to 0° at 1500 s, as shown in Fig. 2d. Equation (13) describes the relationship.

$$v = \begin{cases} v_1, & 500 \leq t < 1500 \\ v_0, & \text{others} \end{cases} \tag{13}$$

where v_1 is 20°.

The last lateral weaving maneuver model aims at modeling a complicated prediction task that traditional methods of trajectory prediction often fail to accomplish. The regulation of bank angle v possesses a 500-s cycle that starts at 300 s, and the value of it reverses during and between each cycle, as shown in equation (14) and Fig. 2f.

$$v = \begin{cases} (-1)^k v_2, & t \in [300 + 500k, 500 + 500k] \\ v_0, & \text{others} \\ (-1)^{k+1} v_2, & t \in [550 + 500k, 750 + 500k] \end{cases} \tag{14}$$

where k stands for the number of control cycles within $[0, 2]$ and v_0 is the weaving baseline. v_2 is set as 30° to differ from v_1.

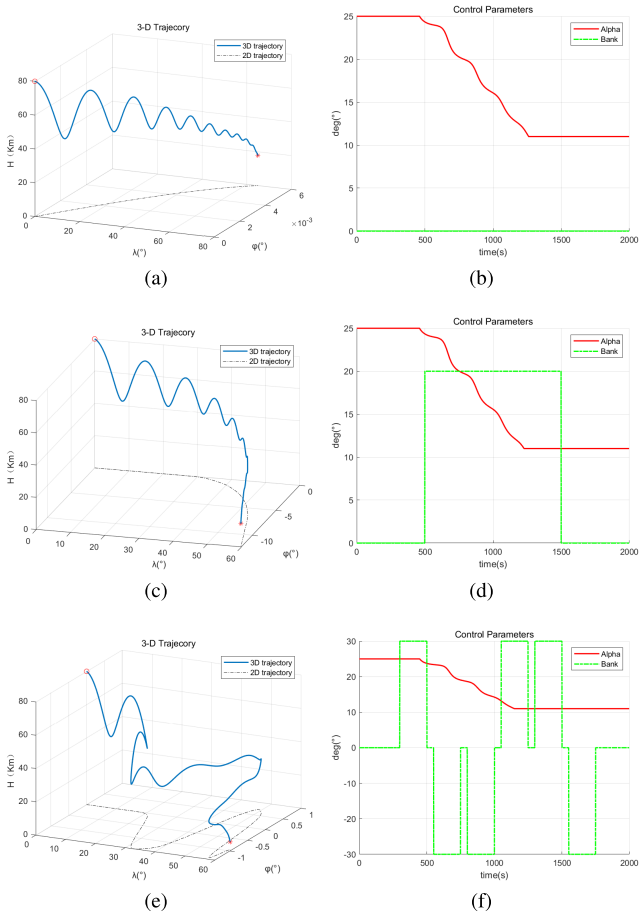


FIGURE 2. The three-dimensional trajectory and control parameters plots of different maneuver models. (a) shows the 3-D trajectory and (b) shows the control parameters plot of the longitudinal-only maneuver model, while (c, d) indicate those of longitudinal with lateral turning maneuver model and (e, f) express those of longitudinal with lateral weaving maneuver model.

C. PROBLEM FORMULATION

Flight trajectory data $\mathbf{X} = [x_1, x_2, \cdots, x_T]$ are represented as a six-dimensional time series that is fully observed in a time order with a constant time interval of one second, where measurement x_T is recorded at time stamp T. Each measurement in the trajectory data is denoted as $x_i = (X_i, Y_i, Z_i, V, \theta_i, \sigma_i)$, where X_i, Y_i, Z_i represent the position of the HGV in the inertial coordinate system of the center of the Earth transformed from the simulation result r_i, λ_i, φ_i, while V_i, θ_i, and σ_i stand for speed, ballistic inclination and course angle, respectively.

When flight trajectory data $\mathbf{X} = [x_1, x_2, \cdots, x_T]$ are observed, we organize these flight trajectory data as several blocks with a tunable data window size w and predict the data of the next second x_{T+1} based on the known block $D_T = [x_{T-w+1}, x_{T-w+2}, \cdots, x_T]$, so the input matrix is $D_T \in \mathbf{R}^{6\times w}$. Likewise, we assume that x_{T+1} that we obtained represents the real data and then predict the future value of x_{T+k+1} based on block $D_{T+k} = [x_{T+k-w+1}, x_{T+k-w+2}, \cdots, x_{T+k}], k \in \mathbf{R}^+$ in a rolling prediction mode.

To comprehensively test the prediction performance of our method, we construct three data sets for three maneuver models described in Fig. 2, while Fig. 2a, Fig. 2c and Fig. 2e show a sample trajectory of each data set. In each data set, there are 2500 trajectories that possess the same maneuver model, while their initial height and initial velocity are set to be different values to distinguish them. For each data set, we adopt 2000 trajectories as the pretraining set and the remaining 500 trajectories as the test set, while 10 percent of the pretraining set for each data set is selected as the validation set. The type of HGV employed in this paper is CAV-L [21], and the values or scopes of initial parameters are listed in Table 1. The values indicate that the initial height is limited within 55-80 km in near space, while the initial velocity is within 4000-6000 m/s, which is less than 20 Ma. All trajectories are simulated with MATLAB R2020a using the dynamic model described in Subsection II-A.

TABLE 1. Trajectory initial parameters.

Parameter	Symbol	value
Initial height	H_0	55-80 km
Initial velocity	V_0	4000-6000 m/s
Initial longitude	λ_0	$0 \times \pi/180$ rad
Initial latitude	ϕ_0	$0 \times \pi/180$ rad
Initial path angle	θ_0	$-0.5 \times \pi/180$ rad
Initial heading angle	σ_0	$90 \times \pi/180$ rad
Mass	m	907 kg
Surface area	S_m	0.4839 m^2

III. METHODOLOGY

In this section, we first present an overview of our proposed architecture DCBNN in Fig. 3 and then introduce those neural network components applied in it. Finally, the strategy of model training is discussed.

A. FRAMEWORK

DCBNN utilizes a nonlinear pattern component and a linear pattern component at the same time, aiming to learn the nonlinearity and linearity of flight trajectory data together. In addition, with the help of the well-designed dual-branch architecture, the nonlinear pattern component can effectively capture the mixture of long- and short-term temporal patterns. Input matrix $D_T = [x_{T-w+1}, x_{T-w+2}, \cdots, x_T]$, where $x_i = (X_i, Y_i, Z_i, V, \theta_i, \sigma_i)$ and X_i, Y_i, Z_i represent the position of the hypersonic vehicle in the inertial coordinate system of the center of the Earth transformed from r_i, λ_i, φ_i, is fed into two parallel channels simultaneously, and the global temporal convolutional layer and local temporal convolutional layer will extract several representation vectors that contain temporal information and dependencies between different variables. These representation vectors enter the corresponding Bi-GRU to further capture the temporal pattern of each vector. At the tail of nonlinear pattern component,

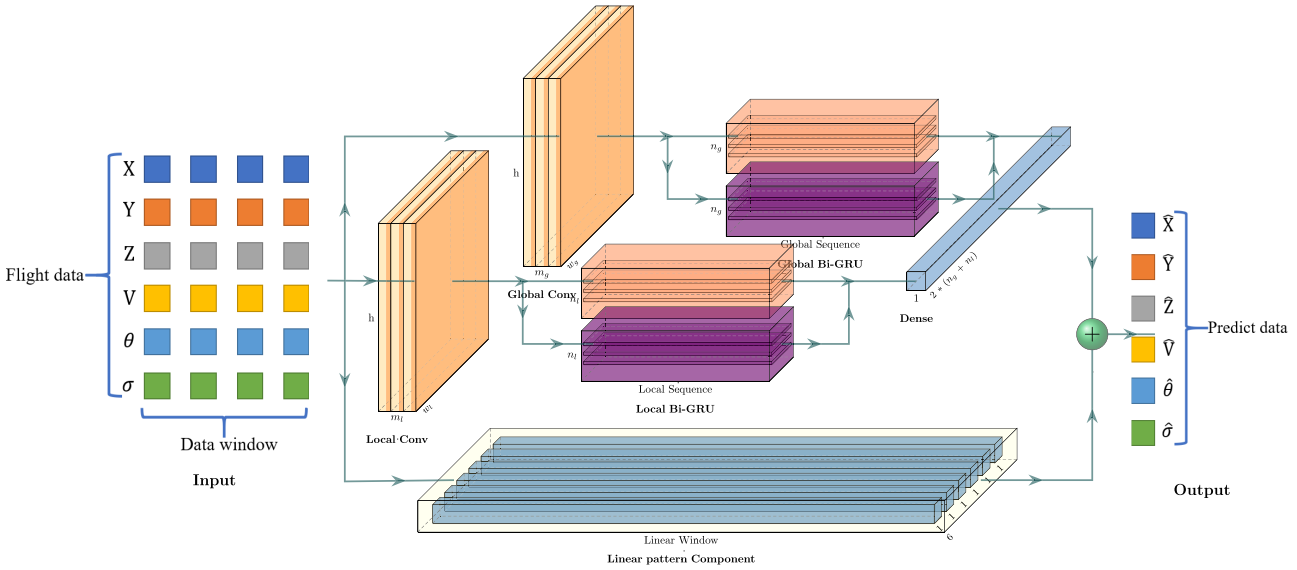


FIGURE 3. An overview of the dual-channel and bidirectional neural network (DCBNN).

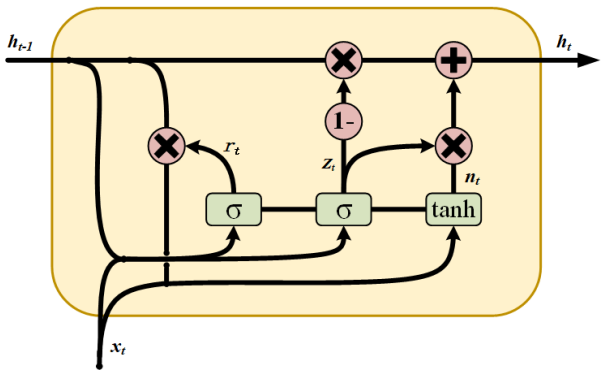


FIGURE 4. Internal structure of GRU cell.

we simply connect the output of the global temporal channel and local temporal channel and pass it through a dense layer to obtain the nonlinear part of prediction result $\hat{y}^n_{T+1}$. Moreover, we construct the linear pattern component with six parallel dense layers that correspond to six-dimensional flight trajectory data and import another input matrix $D'_T = [x_{T-w'+1}, x_{T-w'+2}, \cdots, x_T]$ obtained by cutting D_T off with linear window (w') to obtain the linear part of the prediction result $\hat{y}^l_{T+1}$. Finally, the model generates the final prediction $\hat{y}_{T+1}$ by summing up $\hat{y}^n_{T+1}$ and $\hat{y}^l_{T+1}$.

B. GLOBAL CONVOLUTIONAL LAYER

Previous studies have demonstrated that convolutional neural networks possess outstanding performance in capturing features and performing parallel computing, so DCBNN utilizes a convolutional layer without pooling and activation functions as the first layer of the global temporal channel to preliminarily extract global temporal patterns and dependencies among six variables. This convolutional layer consists of m_g filters with size set as $l_g \times 6$, matching the dimension of the input matrix, where m_g is the number of filters. The k-th filter slides along the time dimension of input matrix D_T and generates

$$c^g_k = W_k * D_T + b_k \tag{15}$$

where $*$ represents the convolutional operation and c^g_k is a vector. W_k and b_k are the weight and bias matrix of this filter, respectively. Each vector produced by a different filter can be considered a hidden representation of input data. By merging all vectors, the output matrix C^g is obtained.

C. LOCAL CONVOLUTIONAL LAYER

Considering the strong correlation of flight trajectory data between short time steps, a local temporal channel is necessary for the model to precisely estimate future trajectory data. Therefore, DCBNN also employs a local convolutional layer as the first layer of the local temporal channel in parallel to the global convolutional layer to focus on modeling local temporal patterns and dependencies among variables. The length of filters used in the local convolutional layer is l_l, which is much smaller than data window w, while the height of filters is still 6. Likewise, each filter will sweep through input matrix D_T and produce a vector. The gathered matrix C^l consists of m_l vectors produced by filters, where m_l is the number of filters in the local convolutional layer.

D. BIDIRECTIONAL GATED RECURRENT UNIT

To prevent the vanishing gradient of traditional RNNs and reduce the computational resources consumed by the LSTM-based model, DCBNN adopts the bidirectional gated recurrent unit (Bi-GRU), a variant of the gated recurrent unit (GRU), which consists of the basic GRU cell illustrated in Fig. 4. The GRU cell is created for the purpose of reducing

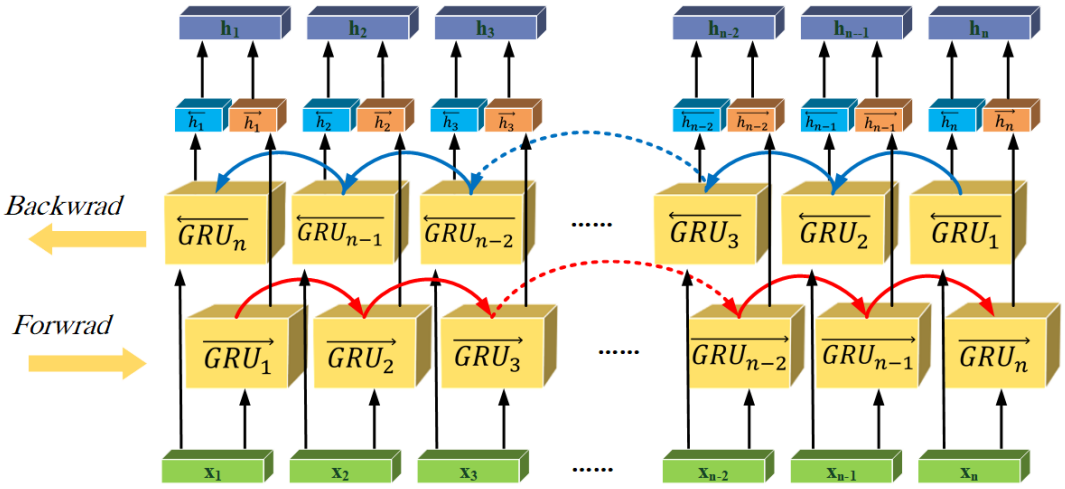


FIGURE 5. Architecture of the Bi-GRU layer.

the number of trainable parameters and simplifying the calculation process of the LSTM cell. It utilizes the reset and the update gate instead of the input, output and forget gate used in the LSTM cell and combines the cell state and hidden state of the LSTM cell into a single hidden state. The calculation process in the GRU cell can be expressed as follows:

$$r_t = \sigma(W_r x_t + V_r h_{t-1} + b_r), \tag{16}$$
$$z_t = \sigma(W_z x_t + V_z h_{t-1} + b_z), \tag{17}$$
$$n_t = \tanh(W_n x_t + V_c(h_{t-1} \otimes r_t + b_n)), \tag{18}$$
$$h_t = (1 - z_t) \otimes n_t + z_t \otimes h_{t-1}. \tag{19}$$

In equations (16) to (19) and Fig. 4 above, r_t and z_t stand for the reset gate and update gate, respectively, n_t denotes the current internal state at time t, while h_t and h_{t-1} represent the hidden state at moment t and $t-1$ respectively. x_t is the input matrix of the GRU cell at moment t, σ and tanh represent *sigmod* and hyperbolic tangent activation functions, respectively, and $\otimes$ is the Hadamard product. Likewise, W_i and V_i represent the weight matrices, and b_i represents the bias vectors.

The function of the reset gate is to determine the forgetting rate of the hidden state at previous moment h_{t-1}, and the impact h_{t-1} possesses on the current internal state n_t is positively correlated with the value of reset gate r_t. The update gate works by controlling the proportion of information passed from the current internal state n_t to the output hidden state h_t. With a higher value of update gate z_t, more information of the current internal state will be preserved. To further strengthen the ability of the GRU cell to capture the temporal pattern of each representation vector, we exploit Bi-GRU in both the global temporal channel and local temporal channel, which not only makes the most of past information but also takes the future information into account. The architecture of the Bi-GRU layer, as shown in Fig. 5, consists of a forward $\overrightarrow{GRU}$ and a backward $\overleftarrow{GRU}$, which produce the hidden state h_t by concatenating hidden states $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ generated by forward and backward GRUs simultaneously. The calculation process can be expressed as equations (20) to (22), where parameter n is the length of the sequence entering the Bi-GRU layer.

$$\overrightarrow{h_t} = \overrightarrow{GRU_t}(\overrightarrow{h_{t-1}}, x_t) \tag{20}$$
$$\overleftarrow{h_t} = \overleftarrow{GRU_{n+1-t}}(\overleftarrow{h_{n-t}}, x_t) \tag{21}$$
$$h_t = concatenate(\overrightarrow{h_t}, \overleftarrow{h_t}) \tag{22}$$

We feed the output matrix of global temporal convolutional layer C^g and local temporal convolutional layer C^l into the corresponding Bi-GRU layer and denote the output of the Bi-GRU at time stamp T of the global temporal channel as h_T^g, while that of the local temporal channel is denoted as h_T^l. Then, a dense layer is used to obtain the nonlinear part of prediction result $\hat{y}_{T+1}^n$ by transforming the combination of h_T^g and h_T^l with the following:

$$\hat{y}_{T+1}^n = WH_T + b, \tag{23}$$

where H_T is the result of concatenating h_T^g and h_T^l, while W and b represent the weight matrix and bias vector respectively.

E. LINEAR PATTERN COMPONENT

As a side effect of the nonlinear nature, convolutional and recurrent layers fail to accurately match the scale of the input and output. This issue obstructs the application of these layers in HGV trajectory prediction, where a small decrease in accuracy will result in thousands of meters of error in position forecasting. Therefore, we introduce a linear pattern component into the structure of DCBNN, which greatly improves the accuracy and robustness of the proposed method. The linear pattern component of DCBNN is composed of six parallel dense layers corresponding to the six dimensions of trajectory data ($x = (X, Y, Z, V, \theta, \sigma)$). The layer number of each dense layer is set to be one, which means that the layer

makes the most of the previous value of a single variable to directly estimate its future value. The calculation of the k-th dense layer can be expressed as

$$\hat{y}_{T+1}^{l,k} = W_k D_T^{'k} + b_k \tag{24}$$

where $\hat{y}_{T+1}^{l,k}$ is the k-th dimension of the linear part of prediction result $\hat{y}_{T+1}^{l}$, $k \in [1, 6]$. Likewise, $D_T^{'k}$ is the k-th dimension of $D_T^{'}$, the input matrix with length w'. W_k and b_k are the weight and bias vector of this layer, respectively.

The final prediction of DCBNN is then obtained by integrating the linear and nonlinear part of the prediction.

$$\hat{y}_{T+1} = \hat{y}_{T+1}^{n} + \hat{y}_{T+1}^{l}$$

F. MODEL TRAINING

1) LOSS FUNCTION

We select the mean square error (MSE) as the loss function of DCBNN in this paper, whose calculation process is shown as follows:

$$L_{MSE} = \frac{1}{k}\sum_{i=1}^{k}(\hat{y}_{T+1}^{i} - y_{T+1}^{i})^2$$

where $k = 6$ is the dimension of trajectory data, and y_{T+1}^{i} is the i-th dimension of y_{T+1}, which is set to be the actual value of trajectory data at the next time stamp x_{T+1}. The MSE loss function amplifies the error between the prediction result and the actual trajectory data with a square relationship, making our model more sensitive to errors.

2) OPTIMIZATION STRATEGY

The optimizer chosen for our method is adaptive moment estimation (Adam) [22] based on adaptive estimation of the low-order moment, which is applied in many machine learning fields (such as natural language processing) as the default optimizer.

3) TRAINING STRATEGY

To improve the performance of prediction while considering the computation time constraints of real-time forecasting, we divide model training into pretraining and retraining parts. During pretraining, a large training data set (simulated or real trajectory) is fed into DCBNN, and the loss function and optimizer above are employed to optimize the parameters of the network iteratively until the validation error of the method converges to a very small number.

This process will consume much time and computation resources, and we finally obtain trained parameters. The retraining process is executed before real-time trajectory prediction, during which we load trained parameters and enter the latest part of the observed trajectory of the target vehicle into DCBNN for a few steps of training. Although the number of retraining steps is limited to confine the computation time, this process still greatly enhances the performance of trajectory prediction.

IV. EXPERIMENTS

In this section, we first conduct an experiment to illustrate the capability of DCBNN to predict trajectory and then analyze the influence of the length of the retraining part. Second, the experiment shows that DCBNN is able to quickly capture the variation in control parameters. Finally, extensive experiments are executed with 7 methods (including our method) on 3 test sets with different maneuver models, demonstrating that our method acquires outstanding performance.

A. EXPERIMENTAL DETAILS

1) IMPLEMENTATION DETAILS

All experiments in this paper are conducted on a computer with an Intel i7-10700 CPU, a GTX2060 super GPU and 16 GB RAM, while methods are implemented using Python 3.7 and PyTorch 1.2.

2) DATA SETTING

We simulate 2500 trajectories for each maneuver model, constructing three separate training data sets and three testing data sets. For each training data set, we randomly select 90 percent of its trajectories to train the proposed model and utilize the remainder as validation sets. As different dimensions of the trajectory may possess different units, nondimensionalization is necessary. Therefore, all trajectory data are normalized using min-max scaling as in equation (25) before implementation in the methods.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{25}$$

3) EVALUATION METRICS

To comprehensively evaluate the performance of trajectory prediction of each method, we employ three conventional evaluation metrics as follows. The first metric is used in many multivariate time series forecasting works [23].

- Root Relative Squared Error (RRSE):

$$RRSE = \frac{\sqrt{\sum_{i,t\in\Omega_{test}}(y_t^i - \hat{y}_t^i)}}{\sqrt{\sum_{i,t\in\Omega_{test}}(y_t^i - mean(y))}} \tag{26}$$

- Mean Error of Velocity (MEV):

$$MEV = \frac{1}{f}\sum_{t=1}^{f}|y_t^4 - \hat{y}_t^4| \tag{27}$$

- Maximal Error of Spatial Distance (MESD):

$$DIST_t = \sqrt{(y_t^1 - \hat{y}_t^1)^2 + (y_t^2 - \hat{y}_t^2)^2 + (y_t^3 - \hat{y}_t^3)^2} \tag{28}$$

$$MESD = \max_{t\in[1,f]} DIST_t \tag{29}$$

where $y_t^i, \hat{y}_t^i$ are the i-th dimension of real trajectory data and prediction result at time t, respectively; (y_t^1, y_t^2, y_t^3) and $(\hat{y}_t^1, \hat{y}_t^2, \hat{y}_t^3)$ denote the actual position and the estimated position at time stamp t, respectively. f is the length of the prediction horizon. The RRSE is a variant of root mean squared error (RMSE) and is designed to produce a more

reliable evaluation of the forecasting performance regardless of the data scale. In contrast, MESD and AEV are designed to demonstrate the performance of trajectory prediction in a physical way. For all these metrics, a lower value is better.

4) METHODS FOR COMPARISON

In comparative experiments, we adopt 6 baselines, including the following methods:

- MLP stands for a multilayer perceptron with four layers, which consists of an input layer, two hidden layers and an output layer.
- CNN represents a neural network consisting of three convolutional layers and two dense layers, and an ReLU function is adopted as the activation function.
- RNN denotes a network composed of a gated recurrent unit (GRU) layer and a dense layer, which forecasts trajectories by separately learning each variable of the trajectory data.
- CNN-L is constructed by combining the CNN method described above with the linear pattern component utilized in DCBNN to demonstrate the influence of the linear pattern component.
- RNN-L is also built by combining the RNN method described above with the linear pattern component utilized in DCBNN. This approach learns each dimension of trajectory data separately, similar to the RNN.
- C-RNN-L is a simplified form of the DCBNN model, which reduces the dual branch of the nonlinear pattern component to a single channel and replaces the bidirectional GRU layer with a traditional GRU layer.

5) EXPERIMENTAL SETTINGS

All methods are optimized by the Adam optimizer, and the loss is calculated using the mean square error. The data window w and linear window w' are set to be 60 and 30, respectively. The length of the prediction horizon is set to be 50 s. For each component of DCBNN, the length of filters used in local temporal convolution is 6, while that of global temporal convolution is 58. The hidden dimension of convolution is chosen from {10, 12, 16}. The hidden dimension of Bi-GRU is chosen from {10, 12, 15}. The epochs of pretraining are set to 100, while the initial leaning rate of pretraining is chosen from {0.001, 0.002, 0.003, 0.004, 0.005} and decreases every 20 epochs with a decrease rate of 0.2. In the retraining process, the number of epochs is 10, and the learning rate is constant at 0.00001. Moreover, we perform dropout in all methods, and the rate is set as 0.3 for pretraining and 0.2 for retraining.

B. TESTING RESULTS

To select the optimal length of trajectory data utilized in the retraining process and obtain an overview of the prediction ability of DCBNN, a series of experiments are conducted by changing the length of real-time trajectory data entered into DCBNN and the start point of forecasting. The length of real-time trajectory data used in retraining is chosen from {0, 100, 150, 200, 250}, where 0 stands for predicting without the retraining process. The prediction start points range from 100 s to 1100 s, with an interval of 100 s. The experimental results of different maneuver models in terms of MESD are shown in Fig. 6. It can be seen from the figure that the best results of three maneuver models are obtained with retraining lengths of 250 s, 250 s, and 200 s, as they possess the lowest mean value of MESD when all start points are taken into consideration. This result confirms that the pretraining and retraining processes can improve the prediction performance of DCBNN. In addition, the prediction without the retraining process obtains the worst result in the data set of the third

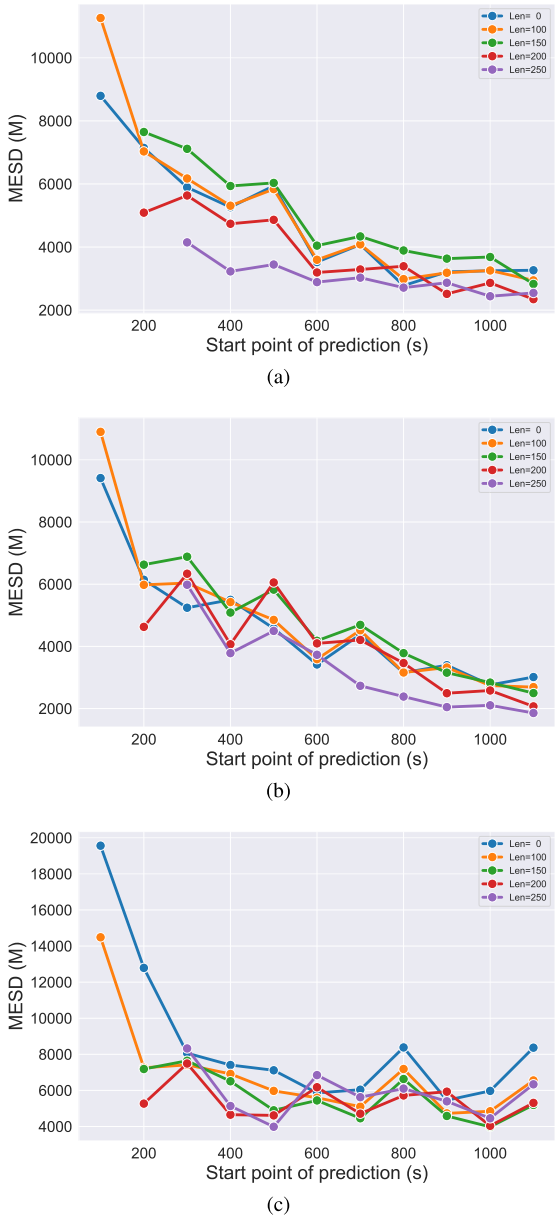


FIGURE 6. MESD of DCBNN with different retraining lengths of three maneuver models. (a) Longitudinal-only maneuver model. (b) Longitudinal with lateral turning maneuver model. (c) Longitudinal with lateral weaving maneuver model.

TABLE 2. The MSE loss of the pretraining process.

Methods	Training epochs				
	20	40	60	80	100
MLP	0.3434	0.0169	0.0066	0.0052	0.0050
CNN	8.3506	8.0586	8.0291	8.0198	8.0294
RNN	1.6285	1.5406	1.5260	1.5223	1.5208
CNN-L	0.0325	0.0154	0.0071	0.0063	0.0061
RNN-L	**0.0179**	0.0087	0.0070	0.0064	0.0062
C-RNN-L	0.0307	0.0195	0.0171	0.0157	0.0154
DCBNN	0.0420	**0.0076**	**0.0041**	**0.0036**	**0.0034**

maneuver model, which further demonstrates that the retraining process is necessary when the target vehicle has a complicated maneuver model. However, it is difficult to directly determine the optimal length of retraining data according to Fig. 6. When the target vehicle has a simple maneuver model, such as longitudinal-only or longitudinal with lateral turning maneuver models, the prediction error will first increase and then decrease as the length of retraining data grows. However, the relationship becomes unclear in forecasting the target using the longitudinal with lateral weaving maneuver model. As a result, we chose 200 s of trajectory data to retrain methods in later experiments since it helps DCBNN achieve a better overall performance.

From Fig. 6a and Fig. 6b, we can also observe that the prediction error gradually decreases as the start point of prediction becomes closer to the end of the flight trajectory. This finding indicates that as the target vehicle encounters friction with the atmosphere, its velocity decreases, making trajectory prediction easier. However, Fig. 6c illustrates that a complex lateral maneuver will greatly increase the difficulty of prediction even when the flight speed is low. This fact demands that a practical method must be capable of rapidly capturing the variation of control parameters.

C. EXPERIMENTS OF CHANGE CAPTURE

From the control parameters plots shown in Fig. 2, we can find that control parameter bank angle v changes two times in the longitudinal with lateral turning maneuver model, while it changes twelve times in the longitudinal with lateral weaving maneuver model. By comparing Fig. 2 and Fig. 6, it is obvious that there are usually peaks of MESD appearing when v changes, except when decreasing to zero. We chose time stamp 500 s in the longitudinal with lateral turning maneuver model and 1050 s in the longitudinal with lateral waving maneuver model, which are both points in time of v change. As the prediction horizon is set to be 50 s and the data window is 60 s, we select time stamps of 50 s ahead and 60 s behind the change point as the test section with an interval of 10 s between each prediction start point. The experimental results are shown in Fig.7.

The prediction results worsen as the start point approaches the v change point, and the worst result is obtained in the vicinity of the change point. These findings demonstrate that

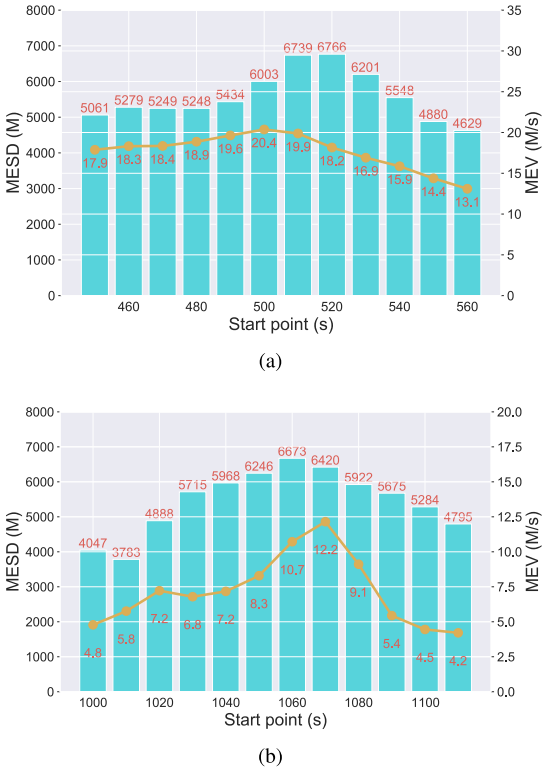


FIGURE 7. MESD and MEV of DCBNN of test sections. (a) Section of the longitudinal with lateral turning maneuver model. (b) Section of the longitudinal with lateral weaving maneuver model.

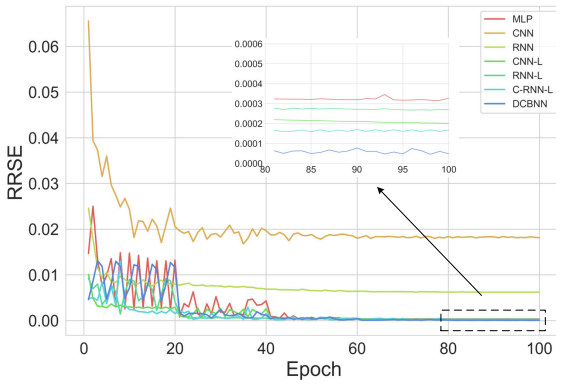


FIGURE 8. RRSE of the pretraining process on the validation set.

DCBNN cannot forecast the alteration of control parameters in the prediction horizon, and prediction errors increase because DCBNN is still using dependencies captured previously to predict the future trajectory. However, soon after the alteration, the prediction errors decrease, which indicates that DCBNN is quickly capturing the new dependencies. This ability enables DCBNN to predict the trajectory of an HGV with complicated maneuver models.

D. RESULTS OF COMPARISON EXPERIMENTS

To highlight the advantage of the proposed DCBNN model in terms of prediction accuracy and robustness, we implement

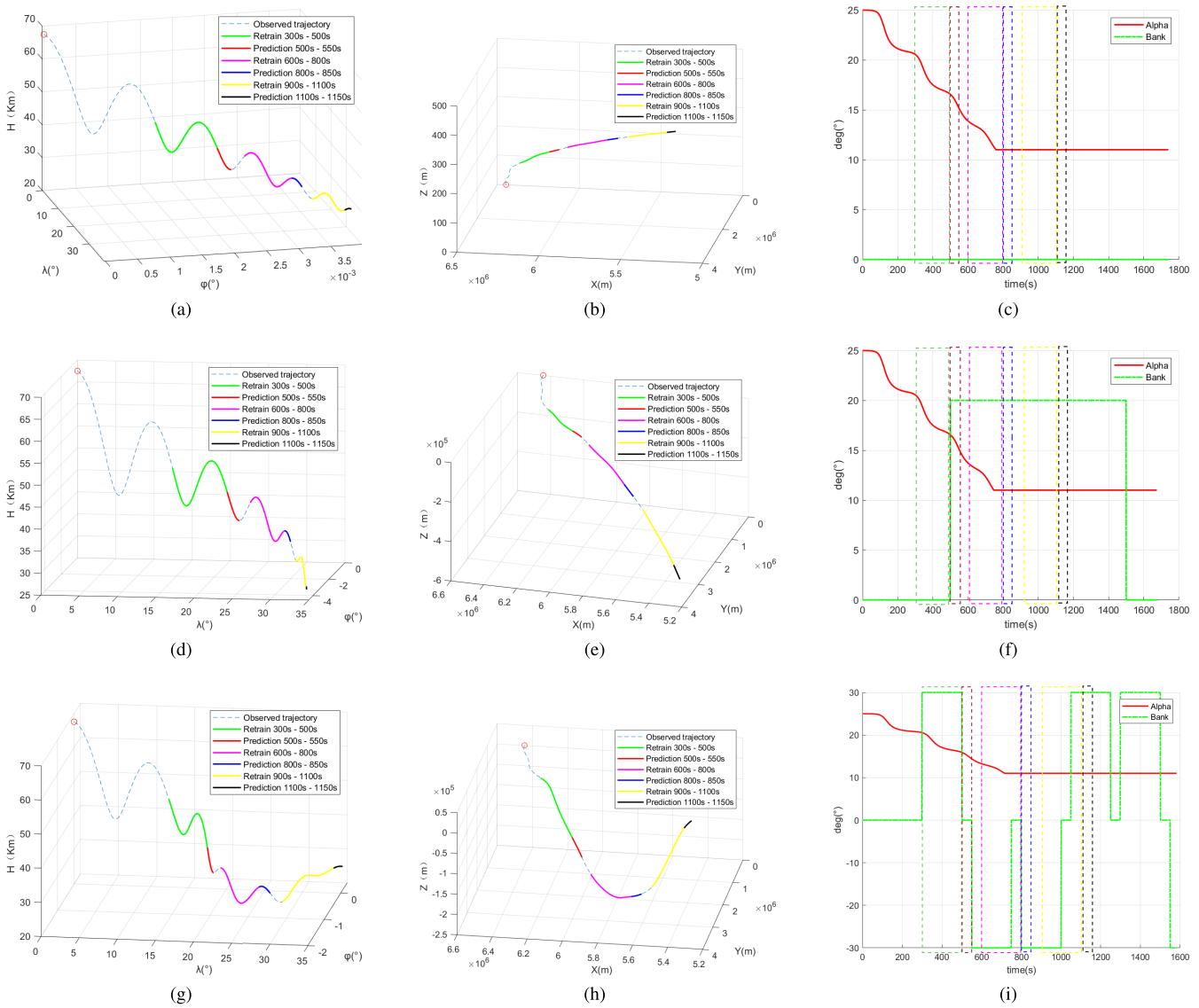


FIGURE 9. The original, transformed 3-D trajectory and control parameters plots of different maneuver models. (a), (b), and (c) shows the retraining and prediction segments in the original 3-D trajectory (a), in the transformed trajectory (b) and control parameters (c) of the longitudinal-only maneuver model, while (d), (e), and (f) indicate those of the longitudinal with lateral turning maneuver model and (g), (h), and (i) express those of the longitudinal with lateral weaving maneuver model.

comparison experiments among DCBNN and six baseline methods on three data sets with different maneuver models. During the pretraining process, we use the MSE loss to evaluate the convergence rate and utilize the RRSE metric to evaluate the performance stability on the validation set. From the MSE loss of the seven methods shown in Table 2, we can easily discover that DCBNN converges to the lowest value from the 40th epoch. Moreover, the values of RRSE of the seven methods are plotted in Fig. 8, indicating DCBNN holds the lowest value after 80 steps of training. These findings demonstrate that DCBNN possesses a fast convergence rate and the best stability.

In each comparison experiment, we adopt 200 seconds of trajectory data for all methods in the retraining process, and the length of the prediction horizon is still set to be 50 s. Three time stamps, namely, 500 s, 800 s, and 1100 s, are selected to be start points of prediction because they are change points or are close to the change point of the bank angle of the longitudinal with lateral weaving maneuver model, while 500 s is also a change point of the longitudinal with lateral turning maneuver model. The example of the original 3D trajectory, the transformed 3D trajectory, and the control parameters of each prediction segment are displayed in Fig. 9.

Table 3 summarizes the evaluation results of all the methods (7) in all the metrics (3), while the values in it are the mean values of the results provided by 500 trajectories in each test set. Each row of the table compares the results of a single method at all prediction segments in the particular metrics,

TABLE 3. Prediction performance comparison by three evaluation metrics.

Data set		Longitudinal Only			Longitudinal with Lateral Turning			Longitudinal with Lateral Weaving		
		Retrain			Retrain			Retrain		
Methods	Metrics	500 s-550 s	800 s-850 s	1100 s-1150 s	500 s-550 s	800 s-850 s	1100 s-1150 s	500 s-550 s	800 s-850 s	1100 s-1150 s
MLP (0)	RRSE	0.080031	0.083063	0.015804	0.042725	0.069700	0.049447	0.050397	0.022997	0.043943
	MESD	51159.94	51616.56	10126.96	28372.59	44667.62	30441.28	31898.69	14291.85	26987.44
	MEV	41.46051	36.95195	10.86416	28.89588	32.11672	20.80589	36.68599	15.63559	20.34447
CNN (0)	RRSE	0.196380	0.200209	0.228033	0.254432	0.261663	0.262293	0.204524	0.363523	0.403809
	MESD	67559.06	83206.01	78473.29	94351.88	94718.13	94426.31	87210.96	142773.47	142372.25
	MEV	82.65617	58.13875	108.64301	91.18072	48.25834	74.90350	87.13183	50.70937	165.16313
RNN (0)	RRSE	0.121455	0.152375	0.162258	0.165953	0.126058	0.259235	0.149120	0.213484	0.218571
	MESD	10200.36	60495.21	61737.51	87628.30	59928.79	103137.77	65333.87	87971.96	82309.25
	MEV	77.66022	84.53931	81.64158	44.51640	64.93204	95.03445	60.83888	52.17848	95.72604
CNN-L (3)	RRSE	0.018308	0.013344	0.008108	0.015198	0.012848	0.021804	0.020958	0.024610	0.047052
	MESD	14074.21	7378.62	4593.98	9151.66	7418.12	12116.76	12864.26	14458.31	27942.04
	MEV	**12.80782**	**7.35182**	6.50145	**9.88482**	10.07310	7.26739	13.63950	9.83311	8.59219
RNN-L (1)	RRSE	0.017892	0.007204	0.009355	0.011538	0.007618	0.006388	0.019409	0.012891	0.015620
	MESD	13136.50	3608.85	5346.39	6616.53	4260.31	3516.53	11485.75	7122.54	9216.52
	MEV	30.40917	10.21354	7.59031	14.39137	8.72661	7.38862	**12.57684**	8.21976	7.82142
C-RNN-L (0)	RRSE	0.013694	0.009111	0.006395	0.014537	0.008900	0.007494	0.017695	0.011360	0.010918
	MESD	7539.31	5095.85	3730.31	8384.23	5159.27	4210.80	10018.31	6294.67	6457.66
	MEV	21.98406	13.36782	6.59165	19.97497	11.16430	7.04883	15.70732	9.75349	14.17081
DCBNN (23)	RRSE	**0.008274**	**0.005541**	**0.003792**	**0.009498**	**0.005582**	**0.003494**	**0.008006**	**0.010195**	**0.008789**
	MESD	**4846.23**	**3333.84**	**2317.17**	**6023.67**	**3387.21**	**2094.71**	**4603.93**	**5734.51**	**5235.09**
	MEV	19.71916	8.34584	**4.39102**	20.52033	**8.59260**	**5.03913**	16.57783	**5.59188**	**4.44576**

and each column represents the results of all methods in a specific prediction segment. The best result for each (data, metrics) pair is highlighted in boldface in this table.

From Table 3, the total value of the boldface results is 23 for DCBNN, 3 for the CNN-L, 1 for the RNN-L and 0 for other comparison methods. Therefore, we first conclude that our proposed model possesses outstanding trajectory prediction performance since it obtains the best results and produces the lowest RRSE and MESE in all prediction segments. This finding also confirms that DCBNN possesses the best robustness. In addition, by comparing the prediction results of CNN and RNN with the results of CNN-L and RNN-L, we discover that there are significant improvements, indicating that by attaching a simple linear pattern component to the model, the nonlinear nature of traditional neural networks is compensated. This phenomenon also demonstrates that the ability of the models to capture linear features of trajectory data will greatly influence its performance of prediction. The fact that DCBNN performs better than CNN-L, RNN-L, and C-RNN-L proves that with the well-designed dual-branch and bidirectional architecture, DCBNN is more accurate and robust in capturing dynamic temporal patterns of trajectories.

V. CONCLUSION

This paper proposes a novel neural network framework, the dual-channel and bidirectional neural network (DCBNN), for the task of hypersonic glide vehicle trajectory prediction in undetectable areas, especially for those trajectories with complex maneuver models where traditional methods often fail to capture the evolution of control parameters. By combining the strength of convolutional, bidirectional recurrent and dense neural networks, we construct a nonlinear pattern component with a dual channel and a linear pattern component with six parallel layers. The statistical results provide evidence that our architecture successfully captures the temporal patterns of trajectories and possesses robustness while facing different maneuver models of the target vehicle.

However, the method proposed in this paper also possesses the following limitations: (*i*) The trajectory prediction is limited to the short term and cannot achieve long-term forecasting. (*ii*) The trajectory data utilized in the retraining process are assumed to be precise; however, observation errors are usually present in actual application scenarios. (*iii*) We divide the training process into a pretraining part and a retraining part for higher prediction accuracy, which introduces more hyperparameters and considerable difficulty in achieving the optimization method performance.

For future research, there are several extending directions worth exploring. First, hyperparameters, such as the filter size and hidden dimension, are mainly tuned based on the validation set. How to automatically choose suitable hyperparameters for DCBNN according to different maneuver models is thus the key to enhancing its practicality. Second, in this paper, we find that noise trajectory data (trajectories with different control parameters within a prediction segment) in the retraining part will reduce the preciseness of forecasting. Therefore, it is necessary to create a method to help determine the length of the retaining part effectively.

REFERENCES

[1] G. Li, H. Zhang, G. Tang, and Y. Xie, "Maneuver modes analysis for hypersonic glide vehicles," in *Proc. IEEE Chin. Guid., Navigat. Control Conf.*, Aug. 2014, pp. 543–548.

[2] Q. Lei, L. Junlong, and Z. Di, "Tracking filter and prediction for non-ballistic target HTV-2 in near space," in *Proc. 27th Chin. Control Decis. Conf.*, May 2015, pp. 3556–3561.

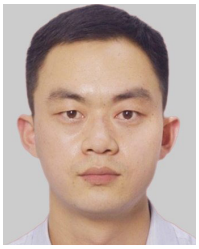
[3] C. Han and J. Xiong, "Method of trajectory prediction for unpowered gliding hypersonic vehicle in gliding phase," in *Proc. IEEE Adv. Inf. Manage., Commun., Electron. Autom. Control Conf. (IMCEC)*, Oct. 2016, pp. 262–266.

[4] Y. Luo, X. Tan, H. Wang, Z. Qu, and T. Li, "Trajectory prediction of hypersonic vehicles based on control quantity prediction," in *Proc. IEEE 4th Inf. Technol., Netw., Electron. Autom. Control Conf. (ITNEC)*, Jun. 2020, pp. 87–91.

[5] P. Lv, S. Liu, W. Yu, S. Zheng, and J. Lv, "EGA-STLF: A hybrid short-term load forecasting model," *IEEE Access*, vol. 8, pp. 31742–31752, 2020.
[6] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain, "Gaussian processes for time-series modelling," *Philos. Trans. Roy. Soc. London A, Math. Phys. Sci.*, vol. 371, no. 1984, 2013, Art. no. 20110550.
[7] T. Niu, J. Wang, H. Lu, W. Yang, and P. Du, "Developing a deep learning framework with two-stage feature selection for multivariate financial time series forecasting," *Expert Syst. Appl.*, vol. 148, Jun. 2020, Art. no. 113237. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417420300634
[8] G. E. P. Box and G. M. Jenkins, "Time series analysis: Forecasting and control," *J. Time*, vol. 31, no. 3, 2010.
[9] W. Ren and M. Han, "Modeling of multivariate time series using variable selection and Gaussian process," in *Proc. 33rd Chin. Control Conf.*, Jul. 2014, pp. 5071–5074.
[10] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.
[11] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.
[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[13] L. Yao and Y. Guan, "An improved LSTM structure for natural language processing," in *Proc. IEEE Int. Conf. Saf. Produce Inf. (IICSPI)*, Dec. 2018, pp. 565–569.
[14] Y. Zhang and X. Lu, "A speech recognition acoustic model based on LSTM -CTC," in *Proc. IEEE 18th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2018, pp. 1052–1055.
[15] L. Ma and S. Tian, "A hybrid CNN-LSTM model for aircraft 4D trajectory prediction," *IEEE Access*, vol. 8, pp. 134668–134680, 2020.
[16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*. [Online]. Available: http://arxiv.org/abs/1412.3555
[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.
[18] N. E. Gaiduchenko and P. A. Gritsyk, "Hypersonic vehicle trajectory classification using convolutional neural network," in *Proc. Int. Conf. Eng. Telecommun. (EnT)*, Nov. 2019, pp. 1–4.
[19] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.
[20] G. Li, H. Zhang, and G. Tang, "Maneuver characteristics analysis for hypersonic glide vehicles," *Aerosp. Sci. Technol.*, vol. 43, pp. 321–328, Jun. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1270963815001091
[21] T. H. Phillips, "A common aero vehicle (CAV) model, description, and employment guide," Tech. Rep., 2003.
[22] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *Comput. Sci.*, 2014.
[23] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, 2018, pp. 95–104.

YANGFAN XIE received the B.E. degree from the School of Electronic and Information Technology, Sun Yat-sen University, where he is currently pursuing the master's degree with the School of System Science and Engineering. His main research interests include artificial intelligence (AI) and trajectory prediction.

XUEBIN ZHUANG received the B.E. degree from the Department of Electronic Engineering, Xiamen University, in 2006, and the Ph.D. degree (Hons.) from the Department of Electronic Engineering, Tsinghua University, in 2011. He is currently an Associate Professor with Sun Yat-sen University. His main research interests include intelligent unmanned systems and hypersonic vehicle technology.



ZEPU XI received the master's degree from the School of Mathematics and Statistics, Lanzhou University, in 2017. He is currently pursuing the Ph.D. degree with the School of System Science and Engineering, Sun Yat-sen University. His research interests mainly include statistical learning theory and reliable machine learning.

HONGBO CHEN received the B.E. and Ph.D. degrees from the Harbin Institute of Technology, in 2001 and 2007, respectively. He is currently a Professor with the Sun Yat-sen University. His main research interests include modeling complex systems and intelligent unmanned systems.