Data-Driven Probabilistic Trajectory Learning with High Temporal Resolution in Terminal Airspace

Jun Xiang* and Jun Chen†
San Diego State University, San Diego, California 92182

https://doi.org/10.2514/1.I011545

Predicting flight trajectories is a research area that holds significant merit. In this paper, we propose a data-driven learning framework that leverages the predictive and feature extraction capabilities of the mixture models and seq2seq-based neural networks while addressing prevalent challenges caused by error propagation and dimensionality reduction. After training with this framework, the learned model can improve long-step prediction accuracy significantly given the past trajectories and the context information. The accuracy and effectiveness of the approach are evaluated by comparing the predicted trajectories with the ground truth. The results indicate that the proposed method has outperformed the state-of-the-art predicting methods on a terminal airspace flight trajectory dataset. The trajectories generated by the proposed method have a higher temporal resolution (1 time step per second vs 0.1 time step per second) and are closer to the ground truth.

Nomenclature

e	=	ego agent
F	=	total number of flights in the current scenario
$\mathcal{F}$	=	set of flights in the current scenario
f	=	one of flight
h	=	output of a hidden layer
p_i^f	=	position of flight f at time i
t_{future}	=	time set for the future trajectory
T^f_{future}	=	ground truth future trajectory of flight f
T^f_{guide}	=	future trajectory guide of flight f
T^f_p	=	past trajectory of flight f
T^f_{pred}	=	generated future trajectory of flight f
t_i	=	time for step i in the trajectory
w^f	=	wind information around flight f
θ	=	learning parameter
ϕ^e	=	context information nearby

I. Introduction

THE terminal airspace is the airspace around airports. This airspace is characterized by low altitudes, multiple agents interacting closely, dynamic conditions, and the need for fast decisions. However, only 4% of all active airports in the United States have control towers that act as a centralized authority for separation [1]. In these nontowered airspaces, pilots bear the sole responsibility for ensuring their aircraft's safety, and the risk of collision is higher. Therefore, predicting the trajectories of other flights in terminal airspace becomes more important.

Furthermore, it is more crucial to track and predict the motion of unmanned aerial vehicles (UAVs) since more and more UAVs are occupying the airspace when the technology for autonomy has evolved and matured. The Federal Aviation Administration (FAA) projects a drone fleet size of 1.81 million units by 2026 [2]. The security service needs to anticipate the misuse of UAVs in public areas, which includes public demonstrations, mass events, and airports [3]. Most of the UAVs should be under surveillance by the unmanned aircraft system traffic management (UTM), which are denoted as in-network UAVs. However, those in-network UAVs could turn into out-network UAVs occasionally due to the limitations and loss of communication. Therefore, timely and accurate trajectory prediction is the key to preventing potential collisions with out-network UAVs. This is even more critical for conflict detection and resolution in UAV landing management [4], which often occurs near crowded airspace [5].

Machine learning (ML) and data-driven methods have demonstrated impressive predictive capabilities across various domains, transcending traditional areas such as motion planning [6], cybersecurity [7–9], and health [10–12]. Furthermore, generative pretrained transformer (GPT) achieves promising results on multiple NLP tasks such as question answering, automatic summarization, and sentiment analysis [13]. Contrastive Language-Image Pretraining (CLIP) dramatically boosts self-supervised learning and achieves promising results on multiple CV tasks, such as image classification [14]. The Vision Transformer (ViT) [15] and its extended work [16] proposed a transformer-based approach for handling computer vision tasks by treating images as sequences of fixed-size patches. By learning patterns and relationships from large amounts of historical data, ML algorithms can generate accurate predictions in real time. These algorithms may be enhanced by incorporating additional information, including weather patterns, air traffic patterns, and other contextual factors that may impact UAV motion [17]. Time Series Modeling [10] demonstrate the superior performance of transformer-based models, particularly PatchTST, in capturing the complex patterns in time series data, and their insights have been crucial in guiding later approaches to time series signal analysis. Three-dimensional trajectory data have many similar features with images and text. The raw trajectory data have the same dimension as the raw image data. Both the trajectory data and text are temporal data. Therefore, data-driven and ML are suitable to be applied in the trajectory prediction of many objects, including aircraft.

In this paper, a learning framework is proposed to provide high-temporal-resolution trajectory prediction while maintaining high accuracy. This paper's primary contributions are as follows:

1) The proposed learning framework can improve long-step prediction accuracy significantly on a terminal airspace dataset, outperforming contemporary state-of-the-art methods.

2) Our work leverages the predictive and feature extraction capabilities of the mixture models and seq2seq-based neural networks while addressing prevalent challenges caused by error propagation and dimensionality reduction.

3) It is the first framework to propose mixture models learn from the output of neural networks to predict flight trajectories. Many researchers have applied neural networks and mixture models together to predict trajectories; however, all the previous methods simply generate

Received 11 September 2024; accepted for publication 16 March 2025; published online 4 April 2025.

*Ph.D. Student, Department of Aerospace Engineering, 5500 Campanile Drive; jxiang9143@sdsu.edu. Student Member AIAA.

†Associate Professor, Department of Aerospace Engineering, 5500 Campanile Drive; jun.chen@sdsu.edu. Senior Member AIAA (Corresponding Author).

the mixture model parameters, including weights, means, and covariances, for each mixture component with a neural network [18,19].

II. Related Work

Trajectory prediction for aircraft is a trending research topic with many proposed methods. Most predictors leverage model-based, cluster-based, and ML-based methods. The most traditional way is predicting future aircraft motion naively by physical equations of motion [20]. OpenAP [21] is a robust aircraft performance model capable of predicting realistic flight trajectories across various flight phases by analyzing the distribution of speed, vertical rates, and altitudes. The 3D hybrid model in [22] combines both data-driven and physics models, so it is capable of predicting UAV motion using minimal real flight data. Recently, the hidden Markov model (HMM) has been utilized to forecast trajectories, taking into account environmental uncertainties [23]. Recurrent neural networks (RNNs) can be trained on historical UAV data to learn the underlying patterns in the motion of UAVs. The trained RNN can then be used to predict future motion based on current position and velocity information. A proposed RNN is designed to forecast aircraft trajectories incorporating weather characteristics [24], which are well suited for time series prediction problems. Moreover, a 4D aircraft trajectory prediction model is proposed in [25], which leverages the deep generative convolutional RNN framework. Generative adversarial networks (GANs), which include a discriminator and a generator, can also generate a flight trajectory [26]. A multilayer perceptron neural network machine learning model predicts the deviations in actual flight paths from the planned routes by extracting features from historical surveillance data that can be generalized for different route structures [27]. A conditional tabular generative adversarial network (CTGAN) [28] model is trained and tested using historical 4D trajectory data from the Hong Kong region. This study [29] compared hybrid recurrent neural network models (CNN-LSTM, CNN-GRU, and SA-LSTM) on aircraft trajectory prediction. Pang et al. [30] used an advanced Bayesian deep learning approach for predicting aircraft trajectories, incorporating the effects of weather.

Predicting the trajectory of airplanes, specifically near terminals, is also a very popular topic. The work in [31] uses a probabilistic generative model to predict flight trajectories near the terminal at KJFK airport by directly learning from aircraft positional data. TrajAir [1] fuses weather data and historical trajectories using an attention-based neural network and generates predictions near KBTP airport. A variational autoencoder (VAE) [32] was adapted to generate random trajectories at the Zurich airport. A Gaussian mixture model (GMM) has been used to detect abnormal aircraft [33] at the Incheon International Airport (ICN). GMM [34] can also predict and simulate trajectories by incorporating clustering analysis near the São Paulo/Guarulhos International Airport (GRU). By using Bayesian networks [35], the statistical patterns of aircraft dynamics can be analyzed to predict their future motion. Most proposed methods only consider a single terminal because applying a general model across different terminals is very challenging due to difficulties in obtaining data, lack of structured representation, and insufficient specificity [36].

Besides aircraft trajectory prediction, many data-driven and ML-based methods are also proposed to predict the trajectory of other objects. In many previous trajectories predicting methods, the GMM is a popular method to represent the distribution of predicted trajectories [19,37,38] because of its capability to model multimodal behaviors of the predicting agents. A proposed extended Kalman filter is used to estimate vessel states, which is subsequently used to help predict the vessel trajectories [39]. The representation of interactions through graphs is also garnering increasing interest, which results in the application of graph neural networks (GNN) for trajectory prediction [40]. An HMM trajectory prediction algorithm is also developed with a self-adaptive parameter that can predict trajectory in a network-constraint environment [41].

In conclusion, many techniques have been proposed for predicting flight trajectories. Nevertheless, most methods, especially feedback-based methods, for predicting flight trajectories are typically very slow and not suited for online tasks [42]. Meanwhile, all the previous trajectory prediction methods still do not meet the accuracy requirement for many applications. Besides, many planning methods struggle to predict the uncertainty of future trajectories. A flight may have multiple different path options, but some predictors cannot anticipate them.

III. Problem Statement

The goal of this paper is to predict the trajectory of flights in the near future using their past trajectories and any useful given information, such as wind information and their neighbors. In this section, the inputs and outputs are going to be defined.

The flight trajectory usually is discretized into multiple waypoints. We locate the physical flight position in airspace with three-dimensional Cartesian coordinates. Considering time, each point is on a 4-D space defined by four variables: time (t), the position in the x-axis (x), the y-axis (y), and the z-axis (z) of the airspace. Usually, z is the altitude of the drone, and (x, y) is the position in the plane that is parallel to the ground. Therefore, the past trajectory of flight f, T_P^f, is expressed as

$$T_P^f = \left\{p_1^f, p_2^f, p_3^f \ldots . p_n^f\right\} \in \mathbb{R}_{n\times 4} \tag{1}$$

$$p_i^f = \left\langle t_i, x_i^f, y_i^f, z_i^f \right\rangle \in \mathbb{R}_4 \tag{2}$$

where $f \in \mathcal{F} = \{1,2,\ldots f,\ldots F\}$ is one of the flights in the current airspace. T_P^f contains points starting from time t_1 to time t_n. The aircraft flies from the position (x_1, y_1, z_1) to the position (x_n, y_n, z_n) in the airspace within the time interval t_1 to t_n. The future trajectory (T_{future}^f) has the same form as the past trajectory.

$$T_{\text{future}}^f = \left\{p_{n+1}^f, p_{n+2}^f, p_{n+3}^f \cdots p_{n+k}^f\right\} \in \mathbb{R}_{k\times 4} \tag{3}$$

The time set in the future trajectory, $t_{\text{future}} = \{t_{n+1}, t_{n+2} \ldots t_{n+k}\}$, is decided by the specific task based on which specific time point of the position is desired to be predicted. Figure 1 shows an example of four agents and their past and future trajectories T_p and T_{future} in the airspace.

The predictor G we proposed should take the past trajectory of the ego agent T_P^e, the context information nearby ϕ^e, and the past trajectory of the flights near the agent $T_P^f, \forall f \in \{1, \ldots, F\}$ as input, and output the ego agent's predicted trajectory T_{pred}^e. Here, the ego

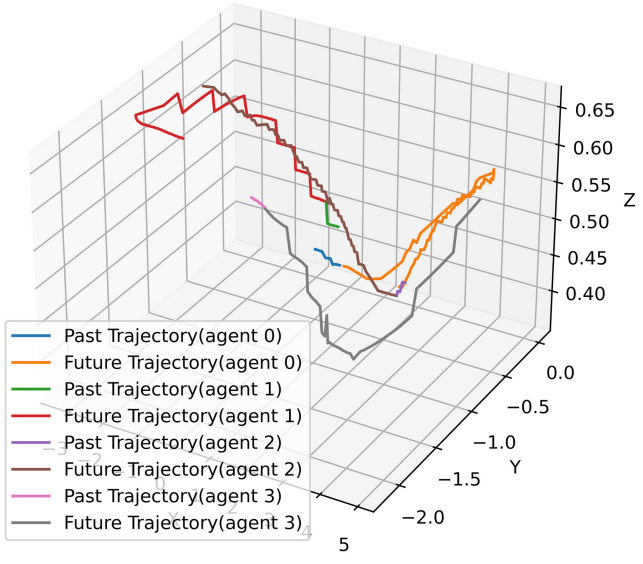

Fig. 1 Past trajectories and future trajectories.

agent is the agent of interest for which we want to predict its future trajectory.

$$T^e_{\text{pred}} = G\left(T^e_P, T^f_P, \phi^e\right) \tag{4}$$

The goal of training the predictor G is to minimize the prediction error Δ between the predicted trajectory T^e_{pred} and the real future trajectory T^e_{future}, as shown in Eq. (5). The detailed performance metric for prediction error in the application will be defined in the Results section.

$$\min_{\theta} \quad \Delta\left(T^e_{\text{pred}}, T^e_{\text{future}}\right) \tag{5}$$

where θ is the learning parameter. Moreover, because the prediction result will be further used for the subsequent tasks, such as collision avoidance, air traffic management, and path optimization, and the flight flies extremely fast, the prediction process should be fast enough so that more time can be left for the subsequent tasks.

$$O(G) \ll t_{n+1} - t_n \tag{6}$$

where $O(G)$ is the computing time of the prediction process, t_n is the time we start to predict, and t_{n+1} is the time when the desired forecast trajectory starts.

In other words, the trajectory predictor should be able to take the past trajectory of the ego agent and all the related context features and predict the future trajectory in a very short time.

IV. High-Temporal-Resolution Prediction Framework

A. Challenges for High-Temporal-Resolution Prediction Framework

High-density autonomy operations require a high-temporal-resolution trajectory prediction model. In statistical words, a mixture model signifies the mixture distribution that represents the probability distribution of observations across the entire population. The mixture model is one of the most promising solutions because the future trajectory of the flights is highly heterogeneous, and a mixture of probabilistic models can be very efficient in representing the future trajectory. Mixture models have been adopted in many trajectory-predicting projects and achieved significant results [31,19]. Compared to the output of deep learning models, the mixture model does not suffer from the same kind of error propagation issues that are often associated with feedforward neural networks or other deep learning models. Figure 2 shows an example of a more recent feedforward neural network model, called TrajAir [1], which has a much larger error when predicting high-temporal-resolution trajectories. The identical neural-network-based predictor performs much worse when generating results for every time step (i.e., TrajAir1 in Fig. 2) compared to generating results for every 10 time steps (i.e., TrajAir10 in Fig. 2). On the other hand, the naive mixture model also presents challenges when predicting trajectories because it relies on context information, where the given features are large-scale and high-dimensional. Dealing with high-dimensional features can be challenging due to the peaking phenomenon [43], which means that the performance of a machine-learning model eventually decreases as the dimensionality of the feature space increases. However, research [44] has shown that the predictive power of the mixture model increases with additional features as long as the net effectiveness of an added feature set exceeds the size of this added feature set. Moreover, although mixture models are capable of clustering and generating high-dimension data, we observed that the 3-D trajectories of the flight generated from high-dimension data by the mixture model lack coherence. Figure 3 shows an example of an incoherent predicted trajectory generated by high-dimension mixture model regression. Even if each point of the generated trajectory is relatively close to the ground truth trajectory, the generated trajectory is not useful. All in all, high-dimensional features are crucial for prediction accuracy, but directly incorporating them into a mixture model proves ineffective. Therefore, in our framework, we proposed the neural network guide-generating module to reduce the dimension of the given information while keeping the useful information as much as possible. The training dataset for the mixture model then only includes three dimensions (x, y, and z in the Cartesian coordinate), avoiding nonsmoothness caused by the projection process.

Therefore, to overcome the above challenges to provide a high-temporal-resolution prediction framework, we propose a learning framework, as shown in Fig. 4, that consists of four main modules: feature extraction, feature fusion, neural networks guide generator, and mixture model predictor. The raw input includes the past trajectory learning models. Information about the ego agent, the past trajectory of other agents, and the context is sent to feature extraction units in the feature extraction modules separately. After the feature extraction units extract important features from the raw input, the feature fusion module combines those features from all the raw input together. The guide generator then generates the trajectory guide using the fused feature, which contains the information from all the inputs. Finally, the mixture model predictor module generates the trajectory prediction based on this guide.

In this paper, we trained a future trajectory predictor with our framework on a general aviation trajectory dataset [1], which uses the past trajectories of the ego agent, the past trajectories of the neighboring agents, and weather content as input to predict the ego agent's future trajectory. We pick neural networks from the TrajAirNet model [1] and probabilistic trajectory models [31] to complete the framework. The details of the implementation of our learning

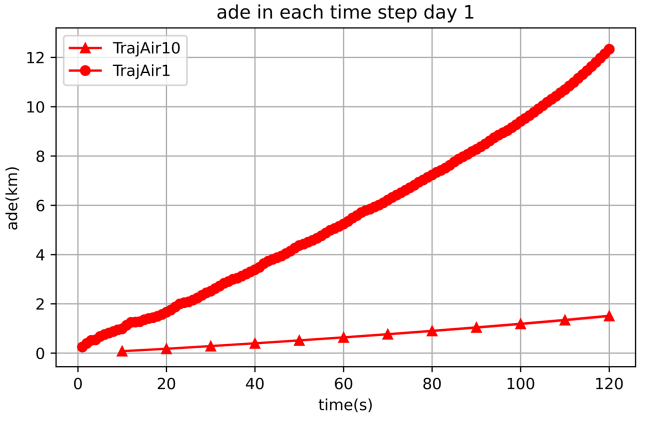

Fig. 2 Feedforward neural networks suffer from error propagation (average displacement error [ADE] increases very fast along the time).

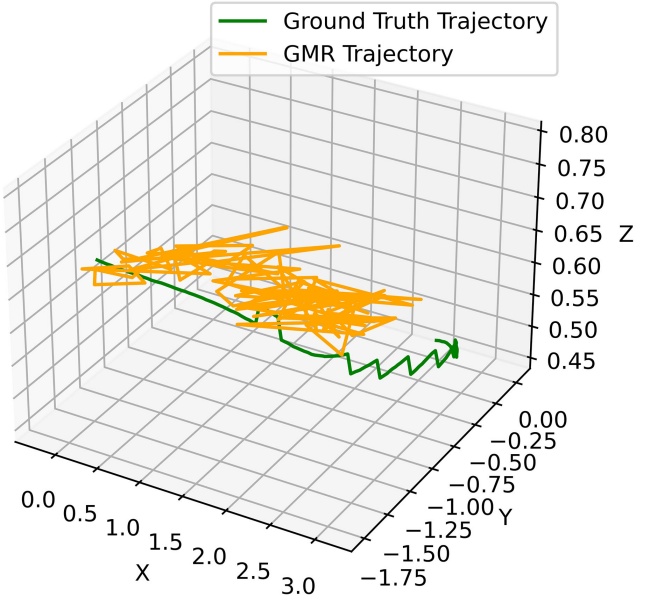

Fig. 3 High-dimensional mixture model regression produces an unsmooth 3-D trajectory.

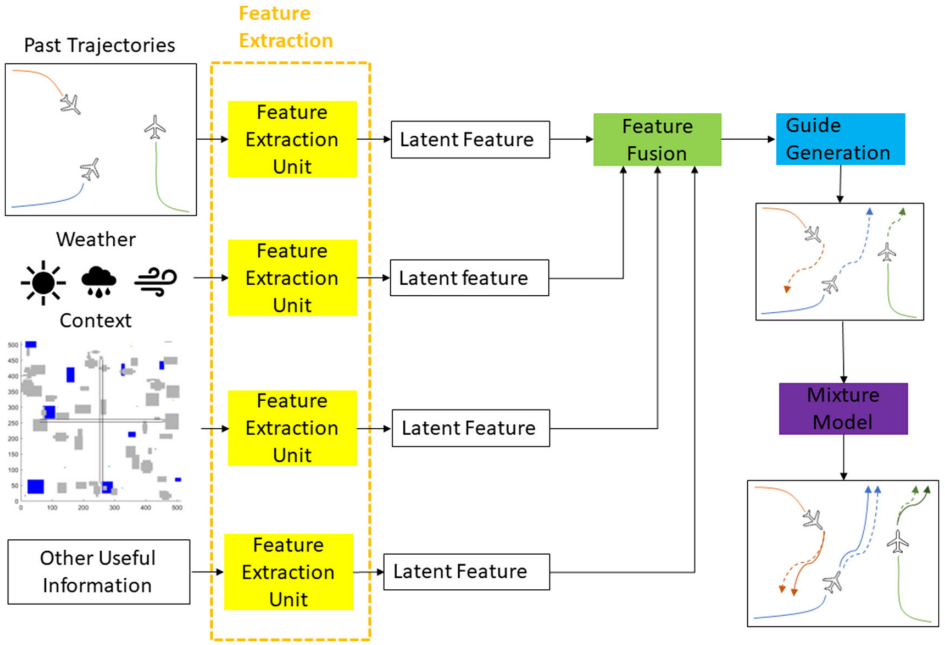

Fig. 4 Proposed learning framework.

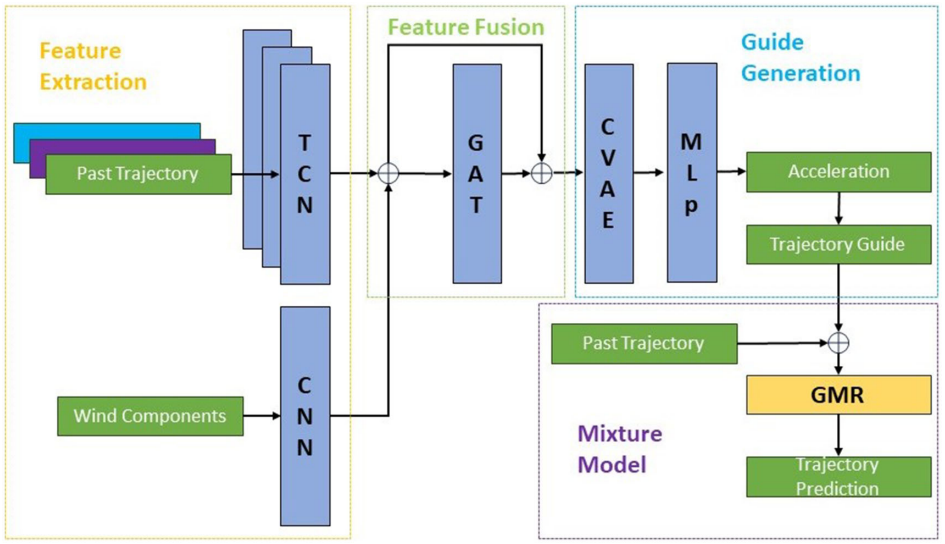

Fig. 5 Implementation details of the framework.

framework are shown in Fig. 5. Each module is introduced in detail as follows.

B. Feature Extraction Module

The feature extraction module transforms all the provided raw data that may affect the future trajectory into a set of features that effectively represent the underlying characteristics of the data and yield improved results compared to applying machine learning directly to the raw data. Feature extraction is a crucial step in the data-driven pipeline. Raw data in different formations need different extraction methods. In this framework, neural-network-based methods are preferable because the extracted feature representation are easier to be integrated into the neural network's guide generator module. In the trajectory predicting tasks, the most commonly provided raw data is the past trajectory, which is a form of sequential data. The long short-term memory (LSTM) networks [45] are useful for extracting features from past trajectories. Another way to extract features from past trajectories is the Autoencoders [46]. Temporal convolutional networks (TCNs) [1] are also applied to handle the past trajectories. Another common raw input is the picture of the environment near the ego agent. Graph neural networks (GNNs) [47] are an option to extract features from the picture for trajectory prediction tasks.

In this paper, the feature extraction module consists of two input layers, which are temporal convolutional networks (TCNs) and convolutional neural networks (CNN). The past trajectories of both ego and neighbor agents will be sent to TCNs, and the wind context will be sent to CNNs. The TCN [48] layers can convert the spatio-temporal trajectory into a latent vector while ensuring that causal connections within the data are preserved. The TCNs simply consist of two layers of dilated causal convolution and rectified linear unit (ReLU), which provide nonlinearity, followed by a spatial dropout.

$$h_{\text{tcn}}^{f} = \text{TCN}\left(T_P^f\right), \quad \forall\, f \in \{1, \ldots, F\} \tag{7}$$

The CNN was widely recognized since [49]. The CNNs can combine the wind context near the different flights together and transfer the feature to a shape we want. The wind near flight f consists of wind velocity in x and y directions.

$$w^f = (w_{vx}, w_{vy}) \in \mathbb{R}_2 \tag{8}$$

Then, in the feature fusion module, the outputs of TCNs and CNNs are concatenated together and become the encoded hidden item.

$$h^f_{\text{enc}} = h^f_{\text{tcn}} \oplus \text{CNN}(w^f) \ \forall\ f \in \{1, \ldots, F\} \tag{9}$$

C. Feature Fusion Module

The feature fusion module combines data from different sources or forms to create a more informative and selective feature representation. It can also combine features from different layers or branches. It is extremely important to notice that the selection of the feature fusion module should be closely aligned with the feature extraction module. Simple concatenation [1] is a traditional feature fusion that is suitable for the same feature space. Pooling [50] also has been used to fuse interaction information in trajectory prediction. Short skip connections fuse the identity mapping features and residual learning features in residual networks (ResNet) [51,52]. Inception, introduced in [53], fuses the outputs of a variety of convolutions and pooling layers. Some research fuses outputs of multiple branches with different kernel sizes with attention [54,55]. At this point, convolutional mechanisms [56] and attention mechanisms are standout neural network structures for feature fusion in trajectory prediction tasks. Many previous papers have applied convolutional mechanisms [57] and attention mechanisms [58] to fuse and extract common information in trajectory prediction tasks.

In this paper, we choose the graph attention network (GAT) [59] as the main neural network in the feature fusion module. The GAT layer can decide which information is more important than others in the encoded hidden item that extract features from the past trajectory and the wind context of all flights. In a scalable way, an attention mechanism, as seen in GATs, can selectively focus on specific agents that affect the trajectory of the ego agent.

$$h^f_{\text{gat}} = \text{GAT}\left(h^f_{\text{enc}}\right) \tag{10}$$

The GATs generate a hidden output for each agent. The hidden output is then concatenated and segregated with the previously encoded hidden item of the ego flight h^e_{enc}.

D. Neural Networks Guide Generator

The neural network guide generator can generate the guide for the mixture probabilistic module, giving fused features from the feature fusion module. After the raw data are extracted and fused together, the processed data will be sent to the guide-generating module. The guide-generating module can be an output layer in the neural networks. Some research [60] simply uses MLP as the output layer to generate the trajectories. Another popular generating output layer is sequential generating, such as the output layer of LSTM [61], RNN [24], and transformer [62]. Variational autoencoders (VAEs) [63] are another type of generative model that generates new data samples that are similar to the ones in the training dataset.

In this paper, we construct a conditional variational autoencoder (CAVE)-based neural network as the guide generator [64]. The concatenated vector from the feature fusion module is sent to CAVE, which is the first layer of the neural network guide generator, as the conditional input. The CVAE consists of two main parts: an encoder $\text{Q}(\cdot)$ and a decoder $\text{P}(\cdot)$. After training, the decoder can take the concatenated vector and a Gaussian noise z and generate a probability distribution. The generated probability distribution then can sample a hidden output for the ego agent h^e_{cvae}.

$$\begin{aligned} z &\sim \mathcal{N}(0, I) \\ h^e_{\text{cvae}} &\sim P\left(h^e_{\text{cvae}} \mid z, h^e_{\text{enc}} \oplus h^f_{\text{gat}}\right) \end{aligned} \tag{11}$$

The output layer is multilayer perceptron (MLP), and the MLP takes the sampled CVAE output h^e_{cvae} and output the acceleration output. The dimension of the output will be $(k/\Delta t, 1)$; Δt is the time interval between each time step; k is the task prediction length. Each element in the output vector represents the acceleration s^e of the ego agent in each time step from t_{n+1} to $t_{n+k/\Delta t}$.

$$s^e_{t_{n+1}:t_{n+k/\Delta t}} = \text{MLP}(h^e_{\text{cvae}}) \tag{12}$$

The acceleration output is subsequently transformed into absolute positions simply with the kinematic equation. And the calculated absolution positions can form the trajectory guide T^e_{guide}.

$$p^e_{t+1} = 2p^e_t - p^e_{t-1} + p^e_t \Delta t^2 \tag{13}$$

$$T^e_{\text{guide}} = \left\{p^e_{n+\Delta t}, p^e_{n+2\Delta t}, \ldots, p^e_{n+(k/\Delta t)\Delta t}\right\} \tag{14}$$

We call the T^e_{guide} trajectory guide because it is then used to guide the mixture model predictor module to generate the prediction trajectory.

E. Mixture Model Predictor Module

The mixture model predictor module is the core module of this framework that will generate the final trajectory prediction given the guide generated by the guide-generating module. In this paper, Gaussian mixture regression (GMR) is deployed as the mixture model predictor module. The GMR predictor is initially introduced in [37] and [65]. The specific one employed in this paper is proposed by [31]. It is one of the most widely recognized trajectory predictors. Before predicting, we need to train a GMM. First, the GMM learns the training dataset through the expectation maximization (EM) [66] algorithm. A trajectory of the training dataset consists of three parts: the first part is the past trajectory T^e_p, the second part is the future trajectory guide generated by the previous neural networks guide generator T^e_{guide}, and the third part is the ground truth future trajectory $\hat{T}^e_F$. After the GMM is trained, the GMM is shown as Eq. (15).

$$p(\boldsymbol{x}, \boldsymbol{y}) = \sum_{k=1}^{K} \pi_k \mathcal{N}_k\left(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\mu}_{xy_k}, \boldsymbol{\Sigma}_{xy_k}\right) \tag{15}$$

where $\mathcal{N}_k(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\mu}_{xy_k}, \boldsymbol{\Sigma}_{xy_k})$ are Gaussian distributions with mean $\boldsymbol{\mu}_{xy_k}$ and covariance $\boldsymbol{\Sigma}_{xy_k}$, K is the number of Gaussians, and x is the input and y is the output, where x is the combined past trajectory and future trajectory guide of the ego agent $(T^e_p, T^e_{\text{guide}}) \in R_{(n+k/\Delta t)\times 4}$, and y is the trajectory prediction $T^e_{\text{pred}} \in R_{k\times 4}$. Note that $\pi_k \in [0, 1]$ are priors that sum up to one. If π_k is larger, the more likely the (x, y) belongs to $\mathcal{N}_k$.

GMR estimates the distributions of the output $\boldsymbol{y}$ with a given input x by determining the conditional distribution $p(\boldsymbol{y} \mid \boldsymbol{x})$, The conditional distribution for each individual Gaussian is defined by

$$\mathcal{N}\left(\boldsymbol{x}, \boldsymbol{y} \mid \boldsymbol{\mu}_{xy}, \boldsymbol{\Sigma}_{xy}\right) \tag{16}$$

The prior and conditional distribution of each individual Gaussian is determined by the following equation:

$$\pi_{\boldsymbol{y}|\boldsymbol{x}_k} = \frac{\mathcal{N}_k(\boldsymbol{x} \mid \boldsymbol{\mu}_{\boldsymbol{x}k}, \boldsymbol{\Sigma}_{\boldsymbol{x}k})}{\sum_{l=1}^{K} \mathcal{N}_l(\boldsymbol{x} \mid \boldsymbol{\mu}_{\boldsymbol{x}l}, \boldsymbol{\Sigma}_{\boldsymbol{x}l})} \tag{17}$$

After the priors are calculated, we can obtain the conditional distribution with the following equation:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \sum_{k=1}^{K} \pi_{\boldsymbol{y}|\boldsymbol{x}_k} \mathcal{N}_k\left(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{y}|\boldsymbol{x}_k}, \boldsymbol{\Sigma}_{\boldsymbol{y}|\boldsymbol{x}_k}\right) \tag{18}$$

Once we have the conditional distribution, we can simply sample an output y from the conditional distribution as the final prediction T^e_{pred}.

F. Loss Function and Accuracy of Trajectory Prediction

The entire neural network pipeline uses integration loss functions

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{traj}} + \mathcal{L}_{\text{cvae}} \tag{19}$$

The $\mathcal{L}_{\text{traj}}$ measures the mean-squared error between the generated guide future trajectory and the ground truth.

$$\mathcal{L}_{\text{traj}} = \text{MSE}\left(T^e_{\text{guide}}, \hat{T}^e_{\text{guide}}\right) \tag{20}$$

The $\mathcal{L}_{\text{cvae}}$ quantifies the KL divergence between the sampling distribution of the latent variable, $\mathcal{N}(0, I)$, and the distribution of the trained latent variable.

$$\mathcal{L}_{\text{cvae}} = D_{kl}\left(Q\left(z \mid \hat{h}^e_{\text{guide}}, h^e_{\text{enc}} \oplus h^e_{\text{gat}}\right) \| \mathcal{N}(0, I)\right) \tag{21}$$

The hidden item $\hat{h}^e_{\text{guide}}$ is the hidden output of TCNs taking the actual ground truth trajectory guide of the ego agent. If the output of the neural networks is closer to ground truth, the KL divergence should be smaller.

$$\hat{h}^e_{\text{guide}} = \text{TCN}\left(\hat{T}^e_{\text{guide}}\right) \tag{22}$$

To measure the accuracy of trajectory predictions, we adopt a suite of three metrics, two of which are used in many previous trajectory prediction works [67], the last one is vertical displacement errors, which pose a greater safety risk in the terminal areas. We use $\hat{p} = \{\widehat{p_t}\}$ to denote the ground truth point, where t is the time we desire to predict the position of the ego flight, and we use $p = \{p_t\}$ to denote the predicted point.

1) Average displacement error (ADE): The ADE is the average L2 norm distance between the points in the ground truth trajectory and the points in the predicted trajectory: $(1/k)\sum_t \|\hat{p}_t - p_t\|_2$ where $t \in t_{n+1}, t_{n+2} \ldots t_{n+k}$, k is the desired time step we want to predict

2) Final displacement error (FDE): The FDE is the L2 norm between the last point in the ground truth trajectory and the last point in the predicted trajectory: $\|\hat{p}_{t_{n+k}} - p_{t_{n+k}}\|_2$

3) Vertical displacement error (VDE): The VDE is the L2 norm height difference between the points in the ground truth trajectory and the points in the predicted trajectory: $(1/k)\sum_t \|\hat{z}_t - z_t\|_2$, where $t \in t_{n+1}, t_{n+2} \ldots t_{n+k}$, k being the desired time step we want to predict.

V. Results

A. Data and Baseline Method Description

The TrajAir dataset [1] is a pioneering collection that captures the multiple flight paths around a typical nontowered airport (Pittsburgh-Butler Regional Airport [ICAO:KBTP]), complemented by the accompanying weather conditions. The FAA has set forth specific guidelines that aircraft must adhere to when entering or departing from nontowered airspace, ensuring the efficiency and safety of all involved parties. Notably, both runways of KBTP follow left traffic patterns. These traffic patterns take a rectangular form, characterized by left-turning maneuvers in relation to landing or takeoff directions. The dataset contains multiple scenes that are formed by multiple frames. Each frame of the dataset contains the frame number, aircraft ID, and aircraft position at x, y, and z dimensions and wind speed at the x and y dimensions.

The evaluations are conducted using a 28-day segment from the TrajAir dataset. This subset encompasses four groups, each containing 7 sequential days from different months. We present the results from our model alongside comparisons with other methods across these four segments. Day 1 includes 7143 predictable agents, day 2 includes 2932 predictable agents, day 3 includes 1936 predictable agents, and day 4 includes 3225 predictable agents. We report the mean, variance, and worst-case values of ADE, FDE, and VDE across all predictable agents. With the aim of creating long-horizon predictions, we use the length of the past trajectory, $n = 11$ s, guide time interval $\Delta t = 10$ s, and predicting length $k = 120$ s. Therefore, the proposed prediction model observes 11 s of the past trajectory and predicts 120 s of the future trajectory with 12 guided points. The proposed prediction model predicts one step every second. The learning rate of the TrajAirNet is 1e−3, and the optimizer is the Adam optimizer. The number of individual Gaussians is 150. We selected TrajAir1 and GMR as the baseline models. All baseline models utilize 11 s of past trajectory data as input to generate predictions for the next 120 s of the future trajectory, similar to the proposed method. Additionally, we present the results for TrajAir10, which generates predictions only for 12 steps (one step every 10 s) and Transformer [68]—a novel closed-loop trajectory predictor, serving as a reference. Note that we report the best result among five samples for each test case and all methods, following some previous studies.

B. ADE, FDE, and VDE Comparison

The ADEs for each baseline method and our method on each dataset are shown in Table 1. The results show that our method yields a considerably lower ADE compared to GMR. The ADE of our method is only 41% on day 1, 58% on day 2, 74% on day 3, and 55% on day 4 in relation to GMR's ADE, while the variance and worst case are also reduced. The Transformer model produces a worse mean result compared to GMR and the proposed method, although it exhibits similar variance and outlier behavior. TrajAir 1 performs significantly worse than all other methods. The ADE of our method is quite close to the TrajAirNet, even though TrajAirNet only generates a 12-point prediction, while our method generates a 120-point prediction.

The FDEs for each baseline method and our method on each dataset are shown in Table 2. Our method also has a markedly lower FDE relative to GMR. Compared to GMR's FDE, our method's ADE is merely 52% on day 1, 69% on day 2, 82% on day 3, and 67% on day 4. Moreover, our method marginally surpasses TrajAir and the transformer in terms of FDE. In the worst-case scenario, the proposed method's FDE is still significantly lower than that of all baseline methods.

The VDEs for each baseline method and our proposed method across the datasets are presented in Table 3. The proposed method achieves excellent VDE performance, largely due to the effective guidance mechanism. Compared to GMR without guidance, our method reduces VDE by approximately 83% on day 1, 79% on day 2, 83% on day 3, and 77% on day 4. At the same time, the Transformer demonstrates poor VDE results, and TrajAir continues to perform poorly as expected. At the same time, the proposed

Table 1 ADE results

ADE (mean ± variance(largest)) unit, km				
Method	Day 1	Day 2	Day 3	Day 4
GMR-net (ours)	0.72 ± 0.68(7.34)	0.87 ± 0.8(8.80)	0.92 ± 0.8(7.10)	0.8 ± 0.78(6.77)
GMR	1.77 ± 1.21(16.69)	1.5 ± 1.11(14.52)	1.24 ± 0.93(15.33)	1.45 ± 1.07(16.57)
TrajAir1	5.57 ± 3.92(27.83)	7.33 ± 8.14(31.45)	12.61 ± 11.73(39.52)	4.68 ± 3.79(25.57)
Transformer	1.82 ± 0.90(9.50)	1.63 ± 0.43(8.65)	2.06 ± 0.70(10.28)	1.81 ± 0.41(9.41)
TrajAir10 (our guide)	0.74 ± 0.64(7.46)	0.87 ± 0.84(8.95)	0.89 ± 0.7(7.05)	0.93 ± 0.87(7.27)

Table 2 FDE results

FDE (mean ± variance(largest)) unit, km				
Method	Day 1	Day 2	Day 3	Day 4
GMR-net (ours)	1.26 ± 1.38(7.22)	1.55 ± 1.59(8.76)	1.69 ± 1.67(6.93)	1.45 ± 1.63(6.81)
GMR	2.44 ± 1.47(15.83)	2.23 ± 1.4(14.15)	2.06 ± 1.34(15.17)	2.18 ± 1.52(15.88)
TrajAir1	12.33 ± 12.33(26.75)	15.55 ± 16.48(29.34)	26.42 ± 22.84(37.19)	10.6 ± 7.73(25.04)
Transformer	2.08 ± 1.60(9.45)	2.12 ± 1.45(9.00)	2.20 ± 1.56(9.45)	2.74 ± 2.47(9.17)
TrajAir10 (our guide)	1.51 ± 1.21(7.32)	1.78 ± 1.66(8.79)	1.76 ± 1.32(7.04)	1.94 ± 1.8(7.27)

Table 3 VDE results

VDE(mean ± variance(largest)) unit, km)				
Method	Day 1	Day 2	Day 3	Day 4
GMR-net (ours)	0.084 ± 0.0078(3.90)	0.099 ± 0.0055(2.40)	0.11 ± 0.016(5.46)	0.10 ± 0.0056(2.88)
GMR	0.48 ± 0.18(11.92)	0.47 ± 0.087(7.11)	0.65 ± 0.27(7.58)	0.43 ± 0.11(7.63)
TrajAir1	3.41 ± 2.21(17.30)	4.41 ± 3.13(19.14)	4.88 ± 3.87(21.21)	3.07 ± 1.97(15.89)
Transformer	0.45 ± 0.032(3.27)	0.30 ± 0.012(3.92)	0.71 ± 0.018(4.23)	0.20 ± 0.0067(3.39)
TrajAir10 (our guide)	0.094 ± 0.0077(3.86)	0.10 ± 0.0055(2.40)	0.11 ± 0.016(5.58)	0.10 ± 0.0056(2.86)

method has lower variance and smaller worst-case outliers compared to others.

The ADE in each time step is shown in Fig. 6. These outcomes indicate that our method offers a more accurate and stable prediction than GMR. Furthermore, the ADE of our method is a little higher than the TrajAir at the first couple of time steps but ultimately overpasses the TrajAir. It means that our method has the same level of accuracy as TrajAir but 10-fold the predicting temporal resolution. Figure 7 shows samples of predictions generated by our method and the baseline methods compared to ground truth. A predictor with lower mean, variance, and outlier values for ADE, FDE, and VDE is more useful for real-world operations. For instance, a better predictor allows for the construction of smaller and more practical reachable sets, which are easier to use in tasks such as collision avoidance and path planning. For example, if the reliable reachable sets of one flight are smaller, the flight can reserve less airspace and leave more airspace for other flights. It is important to note that the TrajAir dataset includes only thousands of trajectories from similar types of agents, with most agents following the same flight plan. As a result, the trained model is likely limited to performing well on this specific dataset.

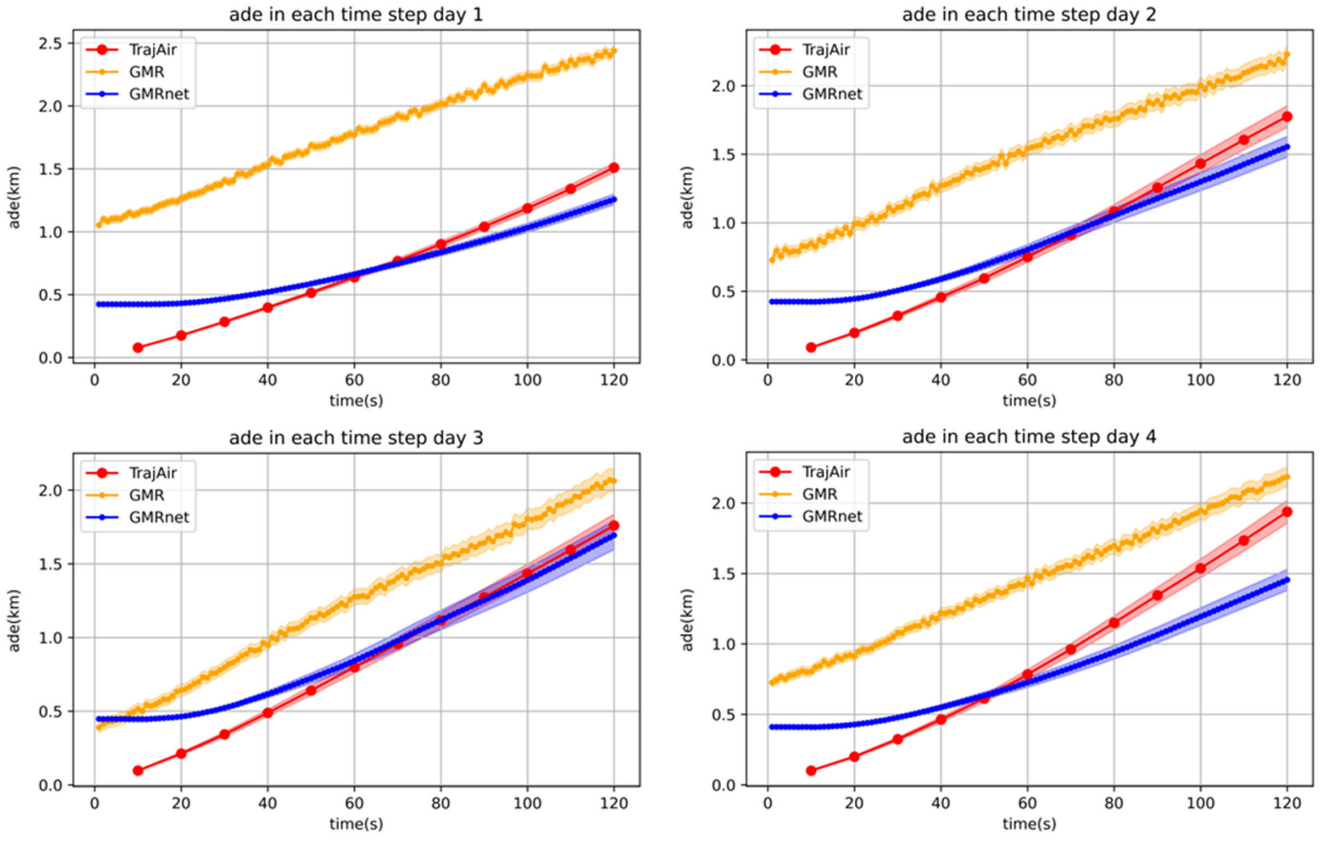

Fig. 6 ADE results for each method.

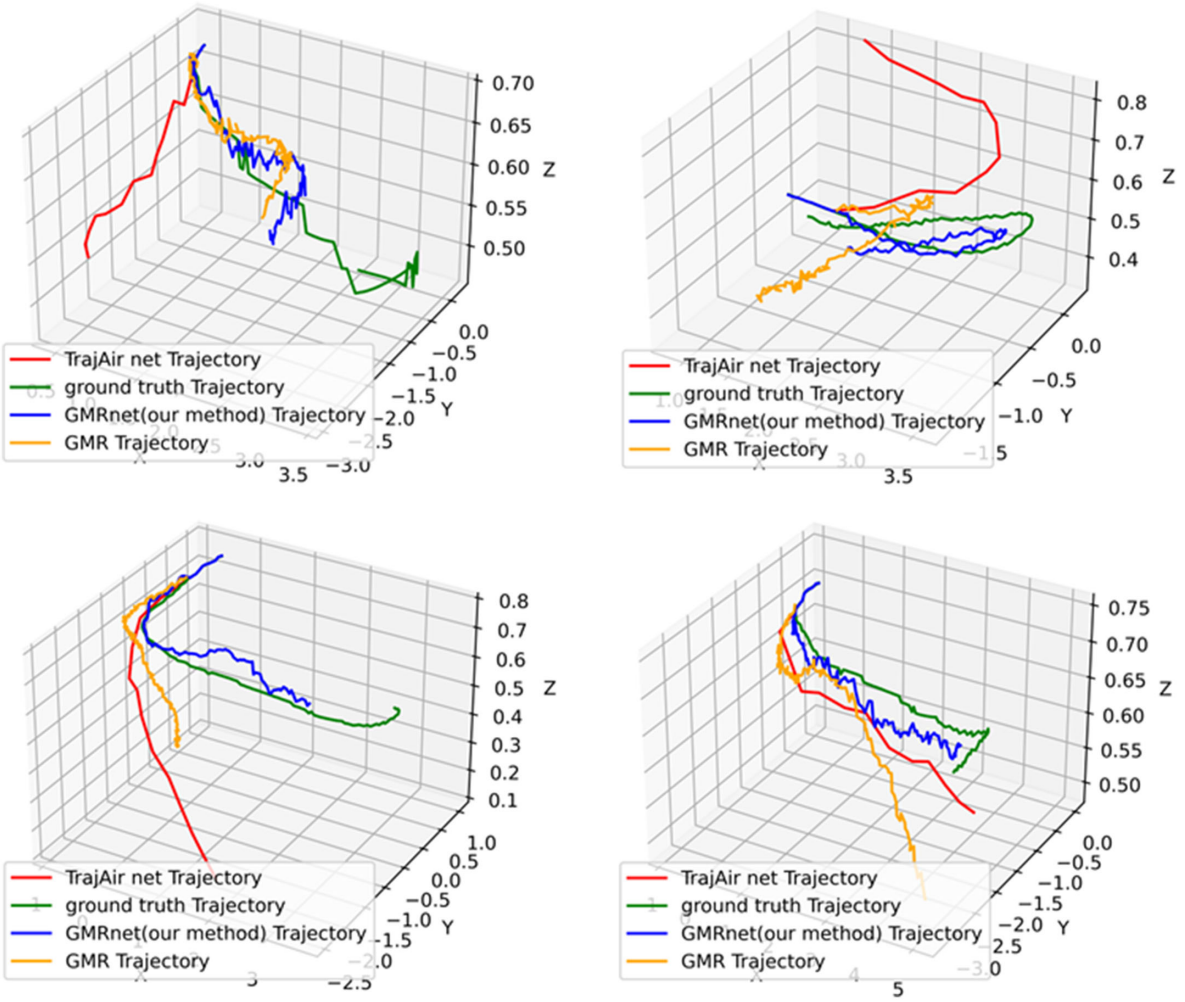

Fig. 7 Examples of prediction comparison for each method.

Table 4 Computing time results

Average generation time per trajectory, s				
Dataset	Day 1	Day 2	Day 3	Day 4
Value	0.234	0.27	0.35	0.31

C. Generation Time Analysis

The result for generation time is shown in Table 4. The result shows that our method is not super-fast because it is a two-phase method, and we run the model on a decent computer. The computing time is still small in magnitude. The generation time for a 1 s prediction only takes less than 0.1 s. The average generation time for a single trajectory is less than 0.4 s, which is significantly shorter than the typical traffic data update interval. This indicates that the proposed method is fast enough for continuous prediction.

VI. Conclusions

This paper proposes a learning framework consisting of four main modules. Each individual module collaborates with each other to achieve the best result because the strength of a module can compensate for the weaknesses of the others. The feature extraction module takes raw data directly, transforming it into more informative features. The feature fusion module can combine extracted features from different sources and formats together. The guide generation module can utilize these fused features, which store all the provided information, and generate a low-dimensional guide for the mixture model module, which finally gives the final trajectory prediction. Each of the modules can be trained to enhance their performance in their respective tasks. We then test the framework by filling the modules with some previous methods. The result shows that our framework outperforms the current state-of-the-art methods on the tested dataset. However, the prediction results are still not sufficient for real-world applications, such as tactical separation tasks. Further research on improving prediction accuracy is required. Meanwhile, similar to most previous methods, we tested the proposed approach on data from a single airport. This does not necessarily guarantee that the method will perform well at other airports with different runway configurations. Therefore, additional datasets from various airports should be evaluated in the future. Given that the proposed method incorporates a fusion mechanism capable of understanding airport configurations, it is expected to outperform most trajectory-based methods. In the future, the current framework is worthy of being exploited. Any novel feature extraction method, feature fusion method, and regression model can empower this framework. For example, the framework may perform better with a stronger context encoder filled in the feature extraction module. Another possible improvement is to apply some diffusion steps on the guidance and generate the final prediction. We also would like this framework to be tested on different trajectory prediction datasets. In this work, we only test the proposed framework on one dataset because there are very few open-source flight trajectory datasets.

Acknowledgment

The authors would like to acknowledge the support from the National Science Foundation under Grant CCF-2402689. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not reflect the views of the National Science Foundation.

References

[1] Patrikar, J., Moon, B., Oh, J., and Scherer, S., "Predicting Like a Pilot: Dataset and Method to Predict Socially-Aware Aircraft Trajectories in Non-Towered Terminal Airspace," *2022 International Conference on Robotics and Automation (ICRA)*, Inst. of Electrical and Electronics Engineers, New York, 2022, pp. 2525–2531. https://doi.org/10.1109/ICRA46639.2022.9811972

[2] FAA, "The Federal Aviation Administration (FAA) Aerospace Forecast Fiscal Years (FY) 2020–2040," 2020.

[3] Laurenzis, M., Rebert, M., Schertzer, S., Bacher, E., and Christnacher, F., "Prediction of MUAV Flight Behavior from Active and Passive Imaging in Complex Environment," *Laser Radar Technology and Applications XXV*, Vol. 11410, April 2020. https://doi.org/10.1117/12.2558561

[4] Wu, P., Yang, X., Wei, P., and Chen, J., "Safety Assured Online Guidance with Airborne Separation for Urban Air Mobility Operations in Uncertain Environments," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 10, 2022, pp. 19413–19427. https://doi.org/10.1109/TITS.2022.3163657

[5] Zhou, Z., Chen, J., and Liu, Y., "Optimized Landing of Drones in the Context of Congested Air Traffic and Limited Vertiports," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 9, 2020, pp. 6007–6017. https://doi.org/10.1109/TITS.2020.3040549

[6] Zhang, Y., Mo, K., Shen, F., Xu, X., Zhang, X., Yu, J., and Yu, C., "Self-Adaptive Robust Motion Planning for High DoF Robot Manipulator Using Deep MPC," *3rd International Conference on Robotics, Artificial Intelligence and Intelligent Control (RAIIC)*, Vol. 2024, 2024, pp. 139–143. https://doi.org/10.1109/RAIIC61787.2024.10671222

[7] Jin, Y., Zhou, W., Wang, M., Li, M., Li, X., Hu, T., and Bu, X., "Online Learning of Multiple Tasks and Their Relationships: Testing on Spam Email Data and EEG Signals Recorded in Construction Fields," *arXiv preprint arXiv: 2406.18311*, 2024, pp. 463–467. https://doi.org/10.1109/AIEA62095.2024.10692685

[8] Weng, Y., and Wu, J., "Leveraging Artificial Intelligence to Enhance Data Security and Combat Cyber Attacks," *Journal of Artificial Intelligence General Science (JAIGS)*, Vol. 5, No. 1, 2024, pp. 392–399. https://doi.org/10.60087/jaigs.v5i1.211

[9] Wu, X., Wu, Y., Li, X., Ye, Z., Gu, X., Wu, Z., and Yang, Y., "Application of Adaptive Machine Learning Systems in Heterogeneous Data Environments," *Global Academic Frontiers*, Vol. 2, No. 3, 2024, pp. 37–50. https://doi.org/10.5281/zenodo.12684615

[10] Ni, H., Meng, S., Geng, X., Li, P., Li, Z., Chen, X., Wang, X., and Zhang, S., "Time Series Modeling for Heart Rate Prediction: From ARIMA to Transformers," *arXiv preprint arXiv: 2406.12199*, 2024, pp. 584–589. https://doi.org/10.1109/EEI63073.2024.10695966

[11] Yukun, S., "Deep Learning Applications in the Medical Image Recognition," *American Journal of Computer Science and Technology*, Vol. 9, No. 1, 2019, pp. 22–26. https://doi.org/10.11648/j.ajcst.20190202.11

[12] Li, X., and Liu, S., "Predicting 30-Day Hospital Readmission in Medicare Patients: Insights from an LSTM Deep Learning Model," *medRxiv*, 2024, pp. 61–65. https://doi.org/10.1109/CBASE64041.2024.10824316

[13] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., "Language Models Are Few-Shot Learners," *Advances in Neural Information Processing Systems*. Vol. 33, 2020, pp. 1877–1901. https://doi.org/arXiv:2005.14165

[14] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al., "Learning Transferable Visual Models from Natural Language Supervision," *International Conference on Machine Learning*, 2021, pp. 8748–8763. https://doi.org/arXiv:2103.00020

[15] Dosovitskiy, A., "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale," *arXiv preprint arXiv: 2010.11929*, 2020, https://doi.org/arXiv:2010.11929

[16] Xin, Y., Du, J., Wang, Q., Lin, Z., and Yan, K., "VMT-Adapter: Parameter-Efficient Transfer Learning for Multi-Task Dense Scene Understanding," *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38, No. 14, 2024, pp. 16085–16093. https://doi.org/10.1609/aaai.v38i14.29541

[17] Chen, Z., and Li, B., "Bridging the Domain Gap: Self-Supervised 3D Scene Understanding with Foundation Models," *arXiv preprint arXiv: 2305.08776*, 2023. https://doi.org/arXiv:2305.08776

[18] Huang, X., McGill, S. G., Williams, B. C., Fletcher, L., and Rosman, G., "Uncertainty-Aware Driver Trajectory Prediction at Urban Intersections," *2019 International Conference on Robotics and Automation (ICRA)*, Inst. of Electrical and Electronics Engineers, New York, 2019, pp. 9718–9724. https://doi.org/10.1109/ICRA.2019.8794282

[19] Shi, S., Jiang, L., Dai, D., and Schiele, B., "Motion Transformer with Global Intention Localization and Local Movement Refinement," *Advances in Neural Information Processing Systems*. Vol. 35, 2022, pp. 6531–6543. https://doi.org/arXiv:2209.13508

[20] Chatterji, G., Sridhar, B., and Bilimoria, K., "En-Route Flight Trajectory Prediction for Conflict Avoidance and Traffic Management," *AIAA Guidance, Navigation, and Control Conference*, AIAA Paper 1996-3766, July 1996. https://doi.org/10.2514/6.1996-3766

[21] Sun, J., Hoekstra, J. M., and Ellerbroek, J., "OpenAP: An Open-Source Aircraft Performance Model for Air Transportation Studies and Simulations," *Aerospace*, Vol. 7, No. 8, 2020, Paper 104. https://doi.org/10.3390/aerospace7080104

[22] Wang, B., Xie, J., Wan, Y., Guijarro Reyes, G. A., and Garcia Carrillo, L. R., "3-d Trajectory Modeling for Unmanned Aerial Vehicles," *AIAA Scitech Forum*, AIAA Paper 2019-1061, 2019. https://doi.org/10.2514/6.2019-1061

[23] Ayhan, S., and Samet, H., "Aircraft Trajectory Prediction Made Easy with Predictive Analytics," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, pp. 21–30. https://doi.org/10.1145/2939672

[24] Pang, Y., Yao, H., Hu, J., and Liu, Y., "A Recurrent Neural Network Approach for Aircraft Trajectory Prediction with Weather Features from Sherlock," *AIAA Aviation Forum*, AIAA Paper 2019-3413, 2019. https://doi.org/10.2514/6.2019-3413

[25] Liu, Y., and Hansen, M., "Indicating Aircraft Trajectories: A Deep Generative Convolutional Recurrent Neural Networks Approach, 2018. https://doi.org/arXiv:1812.11670

[26] Xiang, J., Xie, J., and Chen, J., "Landing Trajectory Prediction for UAS Based on Generative Adversarial Network," *AIAA SCITECH Forum*, AIAA Paper 2023-0127, 2023. https://doi.org/10.2514/6.2023-0127

[27] Zhu, X., Hong, N., He, F., Lin, Y., Li, L., and Fu, X., "Predicting Aircraft Trajectory Uncertainties for Terminal Airspace Design Evaluation," *Journal of Air Transport Management*, Vol. 113, Oct. 2023, Paper 102473. https://doi.org/10.1016/j.jairtraman.2023.102473

[28] Zhang, H., and Liu, Z., "Four-Dimensional Aircraft Trajectory Prediction Based on Generative Deep Learning," *Journal of Aerospace Information Systems*, Vol. 21, No. 7, 2024, pp. 1–13. https://doi.org/10.2514/1.I011333

[29] Schimpf, N., Wang, Z., Li, S., Knoblock, E. J., Li, H., and Apaza, R. D., "A Generalized Approach to Aircraft Trajectory Prediction via Supervised Deep Learning," *IEEE Access*, Vol. 11, 2023, pp. 116183–116195. https://doi.org/10.1109/ACCESS.2023.3325053

[30] Pang, Y., Zhao, X., Yan, H., and Liu, Y., "Data-Driven Trajectory Prediction with Weather Uncertainties: A Bayesian Deep Learning Approach," *Transportation Research Part C: Emerging Technologies*, Vol. 130, Sept. 2021, Paper 103326. https://doi.org/10.1016/j.trc.2021.103326

[31] Barratt, S. T., Kochenderfer, M. J., and Boyd, S. P., "Learning Probabilistic Trajectory Models of Aircraft in Terminal Airspace from Position Data," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 20, No. 9, 2018, pp. 3536–3545. https://doi.org/10.1109/TITS.2018.2877572

[32] Krauth, T., Lafage, A., Morio, J., Olive, X., and Waltert, M., "Deep Generative Modelling of Aircraft Trajectories in Terminal Maneuvering Areas," *Machine Learning with Applications*, Vol. 11, March 2023, Paper 100446. https://doi.org/10.1016/j.mlwa.2022.100446

[33] Choi, H.-C., Deng, C., Park, H., and Hwang, I., "Gaussian Mixture Model-Based Online Anomaly Detection for Vectored Area Navigation Arrivals," *Journal of Aerospace Information Systems*,

Vol. 20, No. 1, 2023, pp. 37–52.
https://doi.org/10.2514/1.I011128

[34] Murça, M. C. R., and de Oliveira, M., "A Data-Driven Probabilistic Trajectory Model for Predicting and Simulating Terminal Airspace Operations," *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, Inst. of Electrical and Electronics Engineers, New York, 2020, pp. 1–7.
https://doi.org/10.1109/DASC50938.2020.9256644

[35] Kochenderfer, M. J., Edwards, M. W., Espindle, L. P., Kuchar, J. K., and Griffith, J. D., "Airspace Encounter Models for Estimating Collision Risk," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 2, 2010, pp. 487–499.
https://doi.org/10.2514/1.44867

[36] Weinert, A., Underhill, N., Serres, C., and Guendel, R., "Correlated Bayesian Model of Aircraft Encounters in the Terminal Area Given a Straight Takeoff or Landing," *Aerospace*, Vol. 9, No. 2, 2022, Paper 58.
https://doi.org/10.3390/aerospace9020058

[37] Fabisch, A., "Gmr: Gaussian Mixture Regression," *Journal of Open Source Software*, Vol. 6, No. 62, 2021, Paper 3054.
https://doi.org/10.21105/joss.03054

[38] Varadarajan, B., Hefny, A., Srivastava, A., Refaat, K. S., Nayakanti, N., Cornman, A., Chen, K., Douillard, B., Lam, C. P., Anguelov, D., et al., "Multipath++: Efficient Information Fusion and Trajectory Aggregation for Behavior Prediction," *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 7814–7821.
https://doi.org/10.1109/ICRA46639.2022.9812107

[39] Perera, L. P., Oliveira, P., and Soares, C. G., "Maritime Traffic Monitoring Based on Vessel Detection, Tracking, State Estimation, and Trajectory Prediction," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 13, No. 3, 2012, pp. 1188–1200.
https://doi.org/10.1109/TITS.2012.2187282

[40] Mo, X., Huang, Z., Xing, Y., and Lv, C., "Multi-Agent Trajectory Prediction with Heterogeneous Edge-Enhanced Graph Attention Network," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 7, 2022, pp. 9554–9567.
https://doi.org/10.1109/TITS.2022.3146300

[41] Qiao, S., Shen, D., Wang, X., Han, N., and Zhu, W., "A Self-Adaptive Parameter Selection Trajectory Prediction Approach via Hidden Markov Models," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 16, No. 1, 2014, pp. 284–296.
https://doi.org/10.1109/TITS.2014.2331758

[42] Xie, G., and Chen, X., "Efficient and Robust Online Trajectory Prediction for Non-Cooperative Unmanned Aerial Vehicles," *Journal of Aerospace Information Systems*, Vol. 19, No. 2, 2022, pp. 143–153.
https://doi.org/10.2514/1.I010997

[43] Hughes, G., "On the Mean Accuracy of Statistical Pattern Recognizers," *IEEE Transactions on Information Theory*, Vol. 14, No. 1, 1968, pp. 55–63.
https://doi.org/10.1109/TIT.1968.1054102

[44] Zollanvari, A., James, A. P., and Sameni, R., "A Theoretical Analysis of the Peaking Phenomenon in Classification," *Journal of Classification*, Vol. 37, No. 2, 2020, pp. 421–434.
https://doi.org/10.1007/s00357-019-09327-3

[45] Chen, X., Zhang, H., Zhao, F., Hu, Y., Tan, C., and Yang, J., "Intention-Aware Vehicle Trajectory Prediction Based on Spatial-Temporal Dynamic Attention Network for Internet of Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 23, No. 10, 2022, pp. 19471–19483.
https://doi.org/10.1109/TITS.2022.3170551

[46] Ivanovic, B., Leung, K., Schmerling, E., and Pavone, M., "Multimodal Deep Generative Models for Trajectory Prediction: A Conditional Variational Autoencoder Approach," *IEEE Robotics and Automation Letters*, Vol. 6, No. 2, 2020, pp. 295–302.
https://doi.org/10.1109/LRA.2020.3043163

[47] Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., and Schmid, C., "Vectornet: Encoding Hd Maps and Agent Dynamics from Vectorized Representation," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Inst. of Electrical and Electronics Engineers, New York, 2020, pp. 11525–11533.
https://doi.org/10.1109/CVPR42600.2020.01154

[48] Bai, S., Kolter, J. Z., and Koltun, V., "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," *arXiv preprint arXiv: 1803.01271*, 2018.
https://doi.org/10.48550/arXiv.1803.01271

[49] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-Based Learning Applied to Document Recognition," *Proceedings of the IEEE*, Vol. 86, No. 11, 1998, pp. 2278–2324.
https://doi.org/10.1109/5.726791

[50] Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A., "Social Gan: Socially Acceptable Trajectories with Generative Adversarial Networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Inst. of Electrical and Electronics Engineers, New York, June 2018, pp. 2255–2264.
https://doi.org/10.1109/CVPR.2018.00240

[51] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Inst. of Electrical and Electronics Engineers, New York, June 2016, pp. 770–778.
https://doi.org/10.1109/CVPR.2016.90

[52] Chen, Z., Jing, L., Liang, Y., Tian, Y., and Li, B., "Multimodal Semi-Supervised Learning for 3D Objects," *arXiv preprint arXiv: 2110.11601*, 2021.
https://doi.org/arXiv:2110.11601

[53] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A., "Going Deeper with Convolutions," *Proceedings of the IEEE Conference On Computer Vision and Pattern Recognition*, Inst. of Electrical and Electronics Engineers, New York, June 2015, pp. 1–9.
https://doi.org/10.1109/CVPR.2015.7298594

[54] Dai, Y., Gieseke, F., Oehmcke, S., Wu, Y., and Barnard, K., "Attentional Feature Fusion," *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, Inst. of Electrical and Electronics Engineers, New York, Jan. 2021, pp. 3560–3569.
https://doi.org/10.1109/WACV48630.2021.00360

[55] Li, X., Wang, W., Hu, X., and Yang, J., "Selective Kernel Networks," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Inst. of Electrical and Electronics Engineers, New York, June 2019, pp. 510–519.
https://doi.org/10.1109/CVPR.2019.00060

[56] Hu, Z., Jia, J., Liu, B., Bu, Y., and Fu, J., "Aesthetic-Aware Image Style Transfer," *Proceedings of the 28th ACM International Conference on Multimedia*, Oct. 2020, pp. 3320–3329.
https://doi.org/10.1145/3394171

[57] Song, X., Chen, K., Li, X., Sun, J., Hou, B., Cui, Y., Zhang, B., Xiong, G., and Wang, Z., "Pedestrian Trajectory Prediction Based on Deep Convolutional LSTM Network," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 6, 2020, pp. 3285–3302.
https://doi.org/10.1109/TITS.2020.2981118

[58] Zhang, K., Zhao, L., Dong, C., Wu, L., and Zheng, L., "AI-TP: Attention-Based Interaction-Aware Trajectory Prediction for Autonomous Driving," *IEEE Transactions on Intelligent Vehicles*, Vol. 8, No. 1, 2022, pp. 73–83.
https://doi.org/10.1109/TIV.2022.3155236

[59] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y., "Graph Attention Networks," *arXiv preprint arXiv: 1710.10903*, 2017.
https://doi.org/arXiv:1710.10903

[60] Zhao, H., Gao, J., Lan, T., Sun, C., Sapp, B., Varadarajan, B., Shen, Y., Shen, Y., Chai, Y., Schmid, C., et al., "Tnt: Target-Driven Trajectory Prediction," *Conference on Robot Learning, PMLR*, 2021, pp. 895–904,
https://doi.org/arXiv:2008.08294

[61] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S., "Social Lstm: Human Trajectory Prediction in Crowded Spaces," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Inst. of Electrical and Electronics Engineers, New York, 2016, pp. 961–971.
https://doi.org/10.1109/CVPR.2016.110

[62] Giuliari, F., Hasan, I., Cristani, M., and Galasso, F., "Transformer Networks for Trajectory Forecasting," *2020 25th International Conference on Pattern Recognition (ICPR)*, Inst. of Electrical and Electronics Engineers, New York, 2021, pp. 10335–10342.
https://doi.org/ICPR48806.2021.9412190

[63] Kingma, D. P., and Welling, M., "An Introduction to Variational Autoencoders," *Foundations and Trends® in Machine Learning*, Vol. 12, No. 4, 2019, pp. 307–392.
https://doi.org/10.1561/2200000056

[64] Kingma, D. P., Mohamed, S., Jimenez Rezende, D., and Welling, M., "Semi-Supervised Learning with Deep Generative Models," *Advances in Neural Information Processing Systems*, Vol. 27, Dec. 2014.
https://doi.org/arXiv:1406.5298

[65] Calinon, S., Guenter, F., and Billard, A., "On Learning, Representing, and Generalizing a Task in a Humanoid Robot," *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, Vol. 37, No. 2,

2007, pp. 286–298.
https://doi.org/10.1109/TSMCB.2006.886952

[66] Dempster, A. P., Laird, N. M., and Rubin, D. B., "Maximum Likelihood From Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, Vol. 39, No. 1, 1977, pp. 1–22.
https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

[67] Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C., Zhou, Y., et al., "Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset," *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Inst. of Electrical and Electronics Engineers, New York, Oct. 2021, pp. 9710–9719.
https://doi.org/10.1109/ICCV48922.2021.00957

[68] Vos, R., Sun, J., and Hoekstra, J., "A Transformer-Based Trajectory Prediction Model to Support Air Traffic Demand Forecasting," *International Conference on Research in Air Transportation*, University of Nanyang Technological University (NTU), Singapore, July 2024, Paper ICRAT-2024.

M. Kochenderfer
Associate Editor