Trajectory Reconstruction and Prediction for Vectored Area Navigation Arrivals

A Thesis in
Air Transportation

By
Seokbin Yoon

Submitted in Partial Fulfillment of the Requirements
for the Degree of Master in Science

Department of Air Transportation
The Graduate School Korea Aerospace University

February, 2024

Trajectory Reconstruction and Prediction for Vectored Area Navigation Arrivals

This certifies that the master's thesis of
Seokbin Yoon is approved

Thesis Commitee

________________ Thesis Supervisor: Prof. Keumjin Lee

________________ Prof. Sang Hyun Kim

________________ Prof. Jay Hoon Jung

The Graduate School
Korea Aerospace University

February, 2024

Contents

Acknowledgements

석사과정을 마치며 지난 2년을 돌아보니 기분이 참 이상함과 동시에 좋습니다. 막연하게 잘 할 수 있을 거라는 생각으로 석사과정을 시작해 여러 어려움을 겪었습니다. 그때마다 많은 분의 도움 및 조언 덕분에 제가 잘 마칠 수 있지 않았나 싶습니다. 이 자리를 빌려 감사의 인사를 전하고자 합니다.

먼저 제게 항상 세심하고 따뜻한 지도를 해주신 이금진 교수님께 첫 번째 감사를 드리고 싶습니다. 지난 2년간의 지도에 대한 감사의 마음을 여기 모두 담기엔 부족하지만, 이 글을 통해 짧게나마 감사의 말씀을 드립니다. 아직 부족한 점이 많지만, 교수님의 가르침을 바탕으로 항상 노력하는 제자가 되겠습니다. 교수님과의 시간을 잘 요약해주는 한 과학자의 말씀을 인용하고자 합니다: ‘특별한 재능이라고는 가지고 있지 않은 평범한 제자에게 스승이 기대와 사랑을 주었을 때, 그가 가진 것에 비해 더 많은 것을 더 이루어낼 수 있다.’ 항상 건강하시길 바라겠습니다. 다시 한 번 감사드립니다.

졸업 논문 심사를 맡아주신 김상현 교수님, 정재훈 교수님께도 깊은 감사를 드립니다. 지도 학생이 아니었음에도 심사 후 주신 귀중한 피드백은 이 논문을 완성하는 데 큰 도움이 되었습니다. 학회 발표 전 리허설을 봐주시고, 복도에서 마주칠 때마다 격려의 말씀을 해주신 김휘양 교수님께도 진심으로 감사드립니다.

ATMCL 연구실 모든 분들께 감사의 말씀을 전합니다. 수영, 형준이형, 별 선배님, 네오미, 홍아 선배님, 입학 초기에 제가 많이 부족했음에도 불구하고 많은 가르침을 주셔서 감사합니다. 선배님들의 미래를 항상 응원하겠습니다. 보름 누나와 유림이에게도 감사의 인사를 전합니다. 같이 입학해서 처음에 모든 게 어려웠고 힘들었지만, 동기가 있어 더 힘을 내고 할 수 있었던 것 같습니다. 항상 응원하겠습니다. 주홍 형님, 주영 형님, 승진, 나현씨에게도 감사의 마음을 전합니다. ATM 연구실 태경이형, 병탁, 문현, 준구에게도

감사합니다. 다른 연구실이었음에도 연구적 교류는 물론, 함께 즐거운 시간을 보낼 수 있었어서 감사드립니다. ATSOL 연구실 현진이형, 주환이, 용석이형에게도 감사합니다. 항상 해주시는 따뜻한 말이 정말 큰 도움이 되었습니다. SMURF 연구실 여러분에게도 감사합니다. 이제 우리 모두 각자의 길을 가게 되겠지만, 각자의 위치에서 열심히 하며 다시 만날 수 있으면 좋겠습니다.

관제교육원 친구들, KUTAM 친구들에게도 감사합니다. 여러분 덕분에 편입 후 학교 생활에 잘 적응하고 여기까지 올 수 있었던 것 같습니다. 2년이라는 짧은 시간이었지만, 여러분과 같이 지낸 학부 생활은 잊지 못할 기억으로 남을 것입니다.

10년간 우정을 함께 쌓아온 범진, 형수, 강호에게도 감사의 말을 전합니다. 여러분과 함께한 시간은 언제나 웃음과 행복으로 가득 차 있었습니다. 앞으로도 서로 싸우지 않고, 변함없는 우정으로 계속 잘 지냈으면 좋겠다.

마지막으로, 항상 저를 믿고 응원해주신 부모님과 할머니, 할아버지께 깊은 감사의 말씀을 드립니다. 제가 공부를 다시 시작한다고 결정했을 때 걱정이 많으셨을텐데도 불구하고 저를 끝까지 믿어주셔서 감사합니다. 더 나은 제가 되어 보답하겠습니다.

저는 이제 더 나은 제가 되기 위해 더 넓고 깊은 실패의 바다로 빠지려 합니다. 많은 어려움이 있겠지만, 지금까지 그래왔던 것처럼 포기하지 않고 앞으로 나아가겠습니다. 끝으로 제 인생의 한 부분이 되어주신 여러분 모두에게 다시 한 번 감사드리며 감사의 글을 마칩니다.

2024년 1월 14일

윤석빈 올림

We will find a way. We always have! - Interstellar

Summary

Trajectory prediction plays a critical role in air traffic management, significantly enhancing operational safety and efficiency. However, there is significant room for improvement in prediction accuracy. Efforts to augment accuracy often leverage artificial intelligence (AI) and other data-driven techniques. A common challenge with these methods is the often insufficient and imbalanced nature of training datasets, which impedes accurate trajectory modeling. This can lead to overfitting, thereby decreasing prediction accuracy. To address these challenges, we propose an approach with an assertion that a trajectory pattern represented by fewer trajectories should not be considered less significant for the prediction model. Given the potential safety implications of incorrect predictions in air traffic operations, it is important to develop a robust prediction model with consistent performance. Our methodology involves the identification of trajectory patterns through clustering with Gaussian Mixture Model (GMM). We then augment the imbalanced training dataset using a trajectory reconstruction model based on the Transformer architecture. A Sequence-to-Sequence (Seq2Seq) architecture with Long Short-Term Memory (LSTM) networks is trained on this augmented dataset. The efficacy of our proposed method is validated using real air traffic data and compared to a model trained with the original training dataset.

List of Tables

List of Figures

1 Introduction

1.1 Motivation

As the demand for air transportation continues to grow, congestion in the airspace surrounding major airports has significantly increased. To manage this congested airspace, various Decision Support Tools (DSTs) have been developed to assist air traffic controllers (ATCs). Central to these tools is a trajectory prediction model, which plays a pivotal role in air traffic management, impacting the capacity, efficiency, and safety of the aviation system. [1, 2, 3].

Historically, trajectory prediction models have relied on aircraft performance models and assumptions of flight intent [4, 5, 6, 7, 8, 9]. While these models are useful, they exhibit limitations, particularly in predicting aircraft trajectories in complex and dynamic air traffic environments influenced by external factors such as weather and airspace situations.

With advancements in Automatic Dependent Surveillance-Broadcast (ADS-B) and flight tracking technologies, there has been a significant increase in available trajectory data. This presents an opportunity to develop more robust and accurate prediction models. Previous research demonstrates the potential of data-driven models in trajectory prediction [10, 11, 12, 13, 14]. These models, learning from historical data, are suitable for capturing the complex interactions between various factors influencing aircraft behavior. However, they often face challenges like insufficient and imbalanced training dataset, which impede effective learning.

Therefore, there is a need to develop trajectory prediction models that are both robust and accurate, capable of effectively managing the diversity and complexity of aircraft trajectory data. Achieving this involves augmenting and balancing the training dataset while preserving its diversity. This can be accomplished through trajectory reconstruction, which generates synthetic trajectories using algorithms or data-driven models. Models trained on such enhanced datasets are likely to exhibit superior performance. Ultimately, this advancement would not only enhance decision-making processes of ATCs but also increase the safety and efficiency of air traffic management, addressing the increasing demands of air traffic and the growing complexity of air traffic operations.

1.2 Thesis Objective

An arriving aircraft generally follows the Standard Terminal Arrival Route (STAR), which is defined as a set of waypoints, or fixed points in three-dimensional space, as illustrated in Fig. 1.1. With the introduction of Area Navigation (RNAV) procedures, aircraft operations along any possible flight path within the coverage of navigation aids are allowed [15]. This provides ATCs with additional airspace for instructing aircraft and a high degree of freedom in managing airspace. Therefore, ATCs can instruct aircraft to deviate from the STARs to more efficiently manage a large number of aircraft by assigning headings, altitudes, and speeds. Such instructions and deviations are referred to as radar vectoring. Therefore, air traffic operations near airports, particularly for arriving air traffic, tend to be influenced by ATC decisions rather than following the STARs. Furthermore, the pattern of radar vectoring varies depending on the specific airspace situation and the preferences or arrival sequencing strategies of ATCs. This variability, in turn, leads to an imbalance in the trajectory dataset.

The imbalance in the trajectory data set pose significant problem in training a data-driven model. A model with the imbalanced training dataset tends to minimize overall loss during training by focusing on the more frequent patterns, thereby ignoring the less frequent patterns [16, 17]. In other words, the data-driven model will adjust its parameters θ mainly according to the most frequent patterns to reduce $J(\theta)$, as formulated in Eq. 1. Consequently, trajectories within less frequent patterns are not learned as effectively as those in more frequent patterns. Given the potential safety implications of inaccurate modeling in air traffic operations, it is crucial to develop a robust data-driven

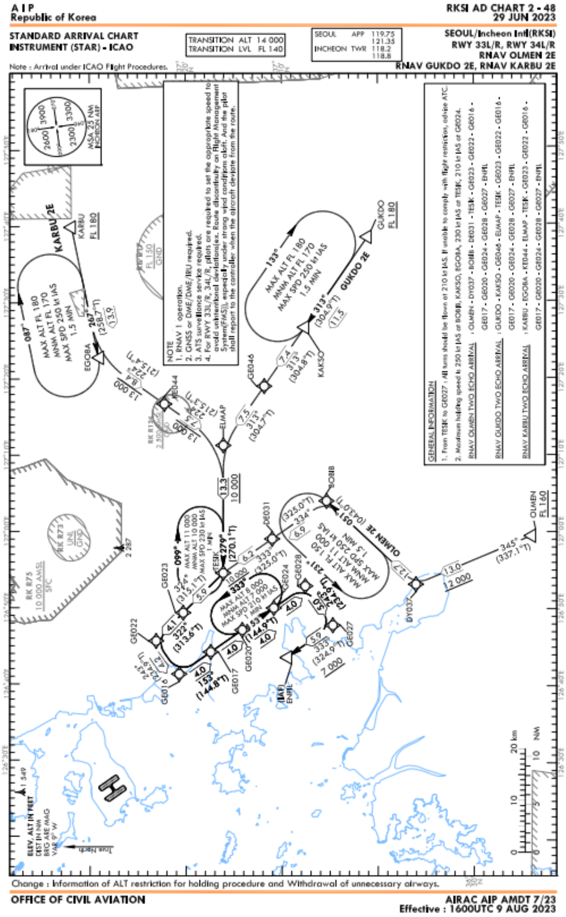


Fig. 1.1: Standard Terminal Arrival Route at Incheon International Airport

model that ensures consistent performance across various trajectory patterns by ensuring a balanced training data set.

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 = \frac{1}{N}\sum_{i=1}^{N}(y_i - \theta \cdot x_i^T)^2 \tag{1}$$

It is widely recognized that the quality of the dataset is crucial when developing data-driven models, often more so than the model (algorithm) itself.

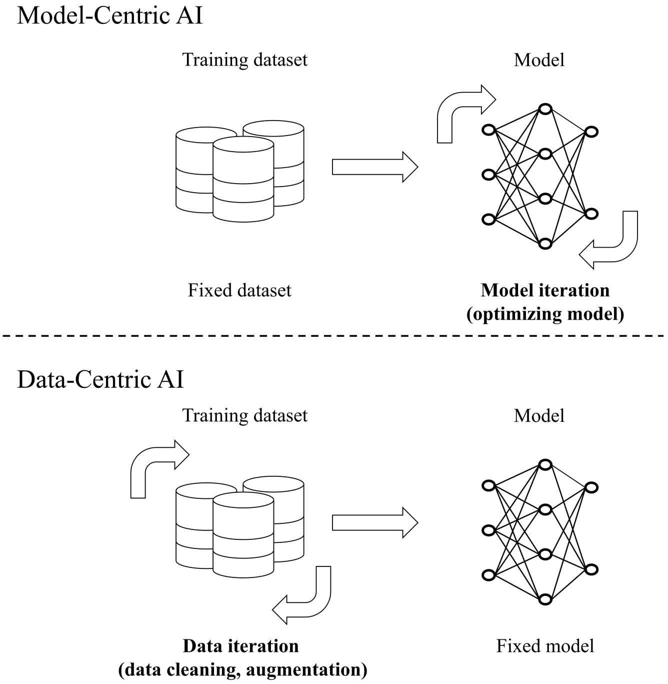


Fig. 1.2: The concept of model-centric AI and data-centric AI

In light of this, a new approach has emerged: Data-centric AI (DCAI) [18, 19]. DCAI is an approach that emphasizes the importance of the quality, preparation, and management of data over the refinement of models. Unlike traditional approaches that often focus heavily on designing and optimizing models (model-centric AI), DCAI prioritizes enhancing the dataset itself, which is illustrated in Fig. 1.2.

In the application of DCAI (i.e., improving the quality of training dataset), it is important to address imbalances within the training dataset [20]. Imbalances can negatively impact the performance of models, especially in the case of underrepresented observations, such as less frequent trajectory patterns in our study. Note that there are various ways to improve the quality of training dataset, including removing outliers and ensuring proper data labeling, in addition to addressing imbalance. In this study, we focus on addressing the imbalance.

To address imbalance in training datasets, techniques such as under-sampling and over-sampling are widely employed [21, 22, 23, 24]. Under-sampling reduces the size of the frequent pattern or dominant class by removing some of its data points, thereby balancing the dataset. In contrast, over-sampling involves augmenting the less frequent or underrepresented class by creating synthetic data points, thus achieving balance in the dataset.

Although both under-sampling and over-sampling are effective methods for addressing dataset imbalances, this study employs over-sampling. The primary objective of this study is to improve overall trajectory prediction accuracy across various trajectory patterns by addressing imbalance. Given that trajectory prediction is a regression task, and considering that aircraft behavior can vary slightly due to the factors such as differences in aircraft and navigation performance, even within the same trajectory pattern, under-sampling data may not be the most suitable approach. Under-sampling could remove important variance needed for precise modeling. This could lead to decreased prediction accuracy for frequent trajectory patterns, while potentially improving it for less frequent ones. This obviously contradicts our objective of this study, which is to improve prediction accuracy for less frequent trajectory patterns, while also maintaining or, if possible, improving accuracy for more frequent trajectory patterns.

In this study, we generate a synthetic trajectory dataset using trajectory reconstruction model to address the imbalance in the trajectory dataset. This

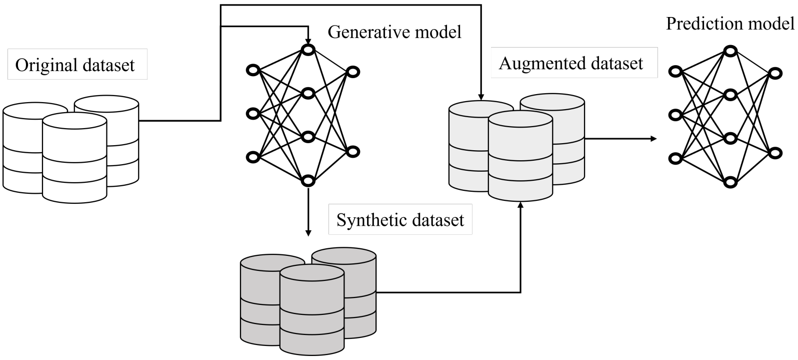


Fig. 1.3: The approach for addressing the imbalance in this study

synthetic dataset is then combined with the existing trajectory dataset to train the trajectory prediction model, as depicted in Fig. 1.3. The primary objective of this study to improve trajectory prediction accuracy by utilizing the generated trajectories as supplemental training data for infrequent trajectory patterns. This approach is preferred rather than collecting additional trajectory data, due to frequent changes in air traffic procedures and aircraft types, which may not reflect the current state of air traffic operations.

1.3 Literature Review

We start by summarizing research papers on trajectory modeling, encompassing trajectory clustering, trajectory reconstruction, and trajectory prediction.

1.3.1 Trajectory Clustering

Clustering, a type of unsupervised learning, automatically divides a dataset into distinct clusters, each characterized by similar features, as exemplified in Fig. 1.4. When applied to trajectories, clustering them into multiple groups is advantageous for analyzing air traffic flow within airspace and for developing DSTs specific to each trajectory pattern. This approach can result in improved performance compared to the use of a single DST for all trajectory patterns. A pattern identification algorithm has been proposed to cluster trajectories based on their representations in a two-dimensional latent space [25]. As trajectory data is inherently high-dimensional, this method effectively reduces the data dimensions and extracts key features for clustering. A framework utilizing hierarchical clustering and Dynamic Time Warping (DTW) was introduced for identifying deeply embedded trajectory patterns in vectored airspace [26, 27]. This approach can identify embedded trajectory patterns, though applying DTW to all trajectories may be time-consuming. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) was used for trajectory pattern identification due to its ability to automatically discover trajectory clusters of any shape, even in the presence of noise and outlier trajectory observations [28, 29]. A framework combining DBSCAN with fuzzy logic for trajectory pattern and flight phase identification was also proposed [30]. Additionally, DBSCAN was used in

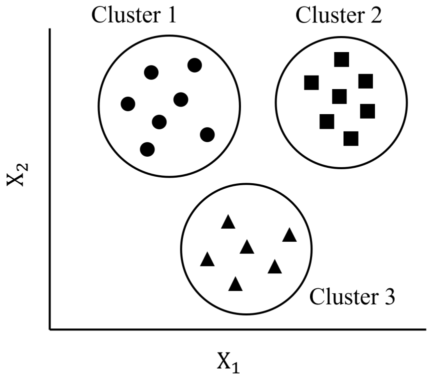


Fig. 1.4: Example of clustering

another study, which also applied kernel density estimation to evaluate the distribution of samples within a cluster [31]. Principal Components Analysis (PCA) and Gaussian Mixture Model (GMM) were utilized, with the GMM being effective under the assumption of normality in nominal flights [32].

1.3.2 Trajectory Reconstruction

Trajectory reconstruction is carried out to generate synthetic trajectories that resemble historical trajectory data. This technique serves two primary purposes: (i) to simulate air traffic operations in existing traffic environments or with new airspace designs and procedures, and (ii) to augment the training dataset for use in other prediction models, subsequently improving their performance. A GMM-based probabilistic trajectory model was developed, capable of generating trajectories and performing inferences in terminal airspace [33]. A framework for generating aircraft trajectories in terminal airspace was proposed [34]. This framework is capable of generating synthetic trajectories by sampling data from a distribution given a new set of procedures. Similarly, [35] performed trajectory reconstruction with feature engineering using a Multi-Layer Perceptron Neural Network (MLP NN). A Temporal Convolutional Variational Autoencoder (TCVAE) was applied for trajectory reconstruction [36]. The authors also tested the generated trajectories in the open-source air traffic simulator BlueSky. Additionally, a Bayesian autoencoder was utilized to identify anomalous behavior in trajectory data [37]. By using the mean and standard deviation of the reconstruction error of generated trajectories, they were able to identify anomalous behavior in the data and determine the most significant features contributing to trajectory reconstruction errors. [38] developed a Gaussian Process-based probabilistic trajectory model, designed for modeling the dispersion of trajectories and generating new trajectories that resemble historical data.

1.3.3 Trajectory Prediction: Data-driven approaches

A simple LSTM-based prediction model was developed [39]. A hybrid architecture based on deep learning was proposed [40]. The architecture combined Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) techniques to extract both spatial and temporal features from aircraft flight trajectory data. In contrast to conventional trajectory prediction approaches that focused on individual aircraft, a novel framework was proposed to predict the entire air traffic scene from a deep learning perspective, employing a ConvLSTM-based autoencoder architecture [41]. A framework for trajectory prediction in terminal airspace was proposed, combining a machine learning-based method and a physics-based estimation method [42]. The collective behavior of flight trajectories was captured through training a machine learning model on historical surveillance data, enabling data-driven predictions for incoming flights. The machine learning predictions were then incorporated as pseudo-measurements in the physics-based estimation algorithm, Residual-Mean Interacting Multiple Models (RM-IMM), to enhance the accuracy of trajectory estimation. The 4D trajectory prediction problem was formulated as a sequence-to-sequence learning problem, and a sequence-to-sequence deep long short-term memory network (SS-DLSTM) was proposed for trajectory prediction [43]. Three kinds of constraints, namely Top of climb, Way-points, and runway direction, were proposed to account for the dynamic characteristics of the aircraft during climbing, cruising, and descending/approaching phases [44]. The model introduced in the paper was designed to maintain long-term dependencies while incorporating these dynamic physical constraints. A Conditional Generative Adversarial Network (CGAN) approach was employed for weather-related aircraft trajectory prediction problems [45]. The paper formulated the problem as predicting trajectories conditioned on the last on-file flight plan and weather effects. A trajectory prediction model based on attention-LSTM was proposed [46]. The model comprised two main stages: the initial stage involved the extraction of time-series features using LSTM, while the second stage incorporated an attention mechanism to process the extracted sequence features. Trajectory prediction with over-sampling technique was proposed to address inherent imbalance in trajectory dataset [47]. By augmenting the dataset with this technique, a notable improvement in prediction accuracy was achieved. Additionally, an Agent-Aware attention mechanism was employed for multi-agent Estimated Time of Arrival (ETA) prediction in [48]. This approach takes air traffic situations into account during the prediction process, resulting in more precise and reliable predictions. [49] proposed a framework that predict

ETA by incorporating probabilistic information for the types of trajectory patterns.

1.4 Thesis Organization

This thesis is organized as follows: Chapter 2 describes the methodology proposed in this study, including trajectory data preprocessing, trajectory pattern identification, trajectory reconstruction, and trajectory prediction, which are detailed in Sections 2.2, 2.3, 2.4, and 2.5, respectively. Numerical experiments conducted to validate the proposed approach are presented in Chapter 3. Finally, the paper concludes in Chapter 4 with a summary of the contributions of our work.

2 Methodology

2.1 Overview

In this subsection, we provide a brief overview of the key components and processes involved in our study. The proposed framework for trajectory reconstruction and prediction consists of three main steps: data preparation, trajectory reconstruction, and trajectory prediction, as illustrated in Fig. 2.1. In the data preparation step, the trajectory data is preprocessed to ensure it is in the proper format for the algorithms used in the following steps. This step includes identifying trajectory patterns and determining the frequency of each pattern based on the number of trajectories it contains. For the less frequent trajectory patterns, the trajectory reconstruction step involves interpolating in the encoded vectors to create a generated trajectory dataset. This synthetic dataset is then combined with the existing trajectory dataset. Lastly, in the trajectory prediction step, the augmented dataset is utilized for training the trajectory prediction model.

2.2 Trajectory Data Preprocessing

In this paper, we use a Automatic Dependent Surveillance-Broadcast (ADS-B) trajectory dataset between January and June 2022 that is collected from FlightRadar24 [50]. This dataset contains aircraft ID, registration, aircraft type, and callsign as part of the flight information, along with latitude, longitude, altitude, heading, speed, and squawk code in the position information. We first convert the geographical coordinates to east-north-up (ENU) coordinates

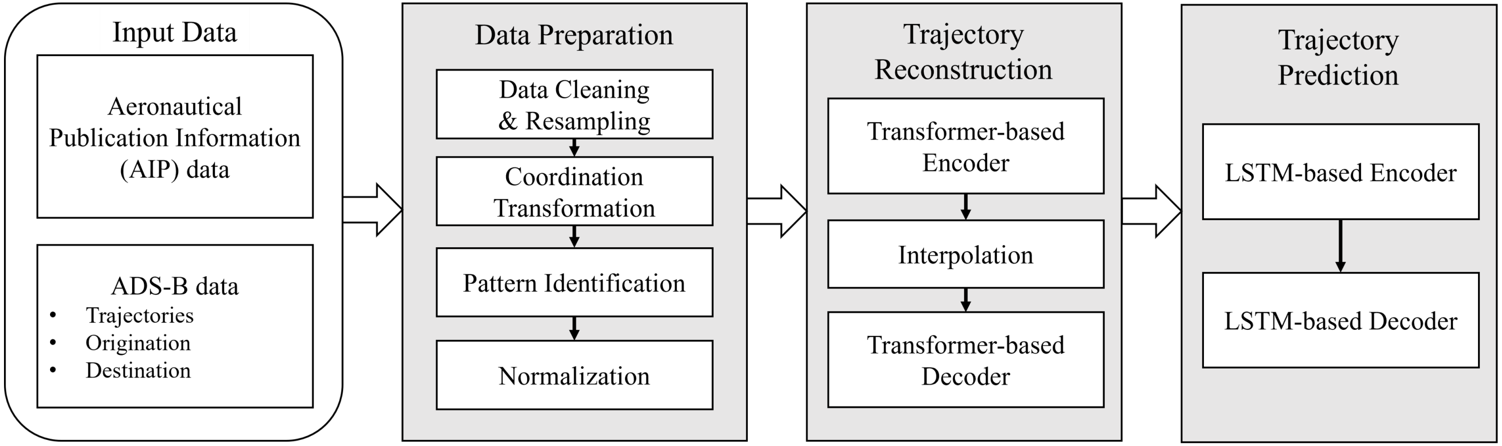


Fig. 2.1: Proposed trajectory reconstruction and prediction framework

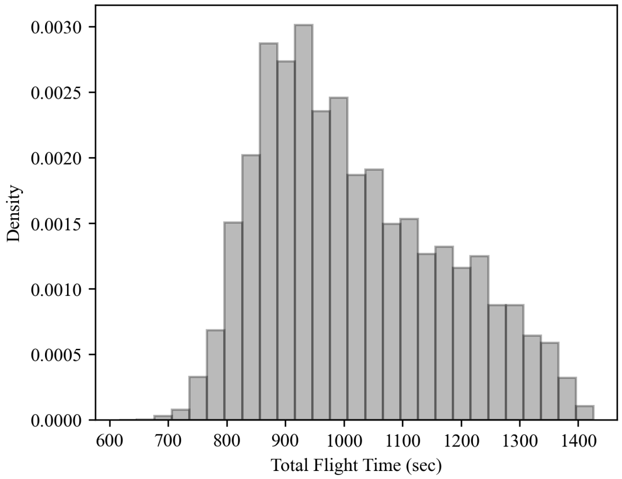


Fig. 2.2: Histogram of total flight time of actual arrival trajectories at Incheon International Airport

determined by the airport reference point (ARP). This transformation simplifies calculations such as distance and heading between points and reduces computational complexity. Next, we extract the trajectory data within a 70-NM radius from the ARP. Due to the irregular sampling rate of ADS-B data, if two consecutive points have a time interval exceeding 3 minutes, we delete the trajectory data, and reconstruct it to have a 4-second sampling rate. Then, we sort the arrival trajectories from the entire dataset based on altitude change and classify the landing runway based on the final position of the trajectory.

Moreover, the trajectory data vary in length, as the total flight time differs, presenting challenges in the direct application of most clustering algorithms, as illustrated in Fig. 2.2. To address this, we interpolate the trajectory data to ensure uniform length across all datasets. For interpolation, we employ the Piecewise Cubic Hermite interpolating Polynomial (PCHIP) [51]. This allows us to interpolate each dimension (east, north, up) in the aircraft trajectory, ensuring all trajectory data have the same fixed length. PCHIP is selected due to its ability to provide a smoother curve compared to other interpolation methods. The PCHIP formulation is as follows:

For every individual segment $x_i \leq x \leq x_{i+1}$,

$$p_i(x) = a_i x^3 + b_i x^2 + c_i x + d_i \tag{2}$$

given the conditions:

$$\begin{aligned} p_i(x_i) &= y_i \\ p_i(x_{i+1}) &= y_{i+1} \\ p\prime(x_i) = y\prime_i &= \frac{y_{i+1} - y_{i-1}}{x_{i+1} - x_{i-1}} \\ p\prime(x_{i+1}) = y\prime_{i+1} &= \frac{y_{i+2} - y_i}{x_{i+2} - x_i} \end{aligned} \tag{3}$$

After resampling the trajectories to a uniform length, we normalize the trajectory data to a range between 0 and 1 using min-max normalization. The normalization process follows the equation:

$$\hat{X} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{4}$$

Here, X represents the original trajectory data, while X_{max} and X_{min} denote the maximum and minimum values of each feature, respectively. The normalization is applied individually for each feature, accounting for their different scales, and $\hat{X}$ is the normalized trajectory data.

2.3 Trajectory Pattern Identification

Given that the trajectory data is unlabeled and lacks distinct identifiers or tags, the direct analysis and identification of air traffic flow, such as trajectory patterns, is challenging. This issue is especially true in vectored area navigation airspace, where trajectory patterns are deeply embedded and identifying the dominant trajectory pattern in this airspace becomes difficult. In this section, we identify trajectory patterns in vectored are navigation airspace using a clustering algorithm.

2.3.1 Gaussian Mixture Model

In this stduy, Gaussian Mixture Model (GMM) is applied to identify trajectory pattern in vectored area navigation airspace. The GMM is a probabilistic model that represents the probability distribution of a dataset as a combination of multiple Gaussian distributions [52], where each Gaussian distribution represents a distinct cluster of data points. One significant advantage of using the GMM as a clustering algorithm is its ability to identify clusters based on their probabilities. This allows the GMM to effectively capture complex trajectory patterns that may not be easily detected by other clustering algorithms. The formulation of the GMM can be expressed as follows:

$$\begin{aligned} p(\boldsymbol{X}) &= \sum_{k=1}^{K} \omega_k \mathcal{N}(\boldsymbol{X}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ \text{subject to} \sum_{k=1}^{K} \omega_k &= 1 \end{aligned} \tag{5}$$

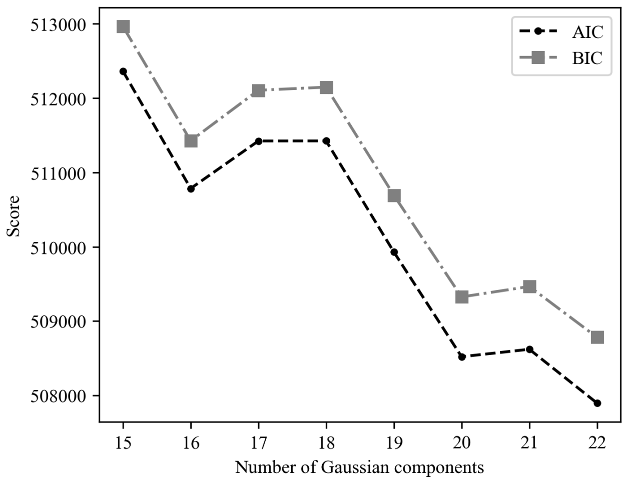


Fig. 2.3: Number of Gaussian components vs. AIC and BIC

Here, $\boldsymbol{X}$ represents a d-dimensional vector, and the probability distribution for a data point $\boldsymbol{X}$ is determined by the weighted sum of K Gaussian distributions. Each Gaussian distribution, denoted by $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, is defined by its mean, $\boldsymbol{\mu}_k$, and its covariance matrix, $\boldsymbol{\Sigma}_k$., which is expressed as:

$$\mathcal{N}(\boldsymbol{X}; \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{\Sigma_k}|}} \exp\left(-\frac{1}{2}(\boldsymbol{X} - \boldsymbol{\mu_k})^T \boldsymbol{\Sigma_k}^{-1} (\boldsymbol{X} - \boldsymbol{\mu_k})\right) \quad (6)$$

The mixture weight of the k^{th} Gaussian distribution is represented by ω_k, and the sum of all of these weights is equal to 1. The parameters of the GMM, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$) for $k = 1, \ldots, K$, are estimated by the Expectation-Maximization (EM) algorithm [53].

To apply the GMM to trajectory data, we vectorize each trajectory data to form a row vector $\boldsymbol{X}$, which can be expressed as:

$$\boldsymbol{X} = [\mathbf{x}_1^1, \mathbf{x}_1^2, \ldots, \mathbf{x}_1^f, \ldots, \mathbf{x}_t^1, \mathbf{x}_t^2, \ldots, \mathbf{x}_t^f] \quad (7)$$

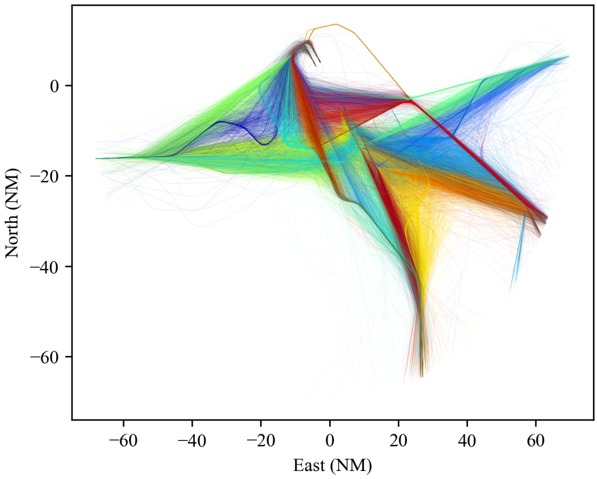


Fig. 2.4: Pattern identification result by GMM

where t represents the final time step of the trajectory, f represents the feature dimension (i.e., east, north, and up), and $\mathbf{x}_t^f$ denotes the value of feature f at time step t. The number of clusters K (i.e., the number of Gaussian components) and was chosen based on the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) [54, 55], as illustrated in Fig. 2.3. The AIC evaluates model fit and complexity by rewarding the likelihood of the model (i.e., how well the GMM fits the given data $\boldsymbol{X}$) and penalizing based on the number of parameters, with a fixed penalty of twice the parameter count. The BIC also assesses model likelihood but applies a penalty for complexity that increases with the logarithm of the sample size. Fig. 2.4 depicts the pattern identification results applied to the actual trajectory dataset. This figure confirms that distinct trajectory patterns can be successfully identified.

2.4 Trajectory Reconstruction

In this section, we address the imbalance issue within the trajectory dataset by reconstructing trajectories from less frequent trajectory patterns. Trajectory reconstruction can be approached either through algorithms (i.e., over-sampling) or data-driven models. A well-known over-sampling algorithm is the Synthetic Minority Over-sampling Technique (SMOTE), which generates new samples by interpolating between k-nearest neighbors [23]. However, SMOTE has limitations in trajectory reconstruction: (i) the generated trajectory data may not reflect the true underlying function of actual trajectory data, such as aircraft performance; (ii) SMOTE's implementation does not consider the temporal relationships within the time-series data, which is critical in trajectory reconstruction.

To address these challenges, we leverage the concept of an Autoencoder (AE) that learns efficient representations of input data [56, 57], along with a Transformer [58]. The AE comprises an encoder, denoted as $\mathrm{Enc}(\boldsymbol{X}; \theta)$, which compresses the input data $\boldsymbol{X}$ into a latent representation $\mathcal{Z}$, and a decoder, denoted as $\mathrm{Dec}(\mathcal{Z}; \phi)$, for reconstructing the output data $\hat{\boldsymbol{X}}$. Here, θ and ϕ represent the parameters of the encoder and decoder, respectively. These properties enable the AE to effectively learn and replicate the underlying function of the data, thus generating new samples that closely resemble the original data. Integrating the concept of AE with the Transformer allows us to generate new trajectories that effectively address the two aforementioned challenges. Fig. 2.5 presents a comparison between the conventional approach and our proposed approach. Further details are discussed in the following section.

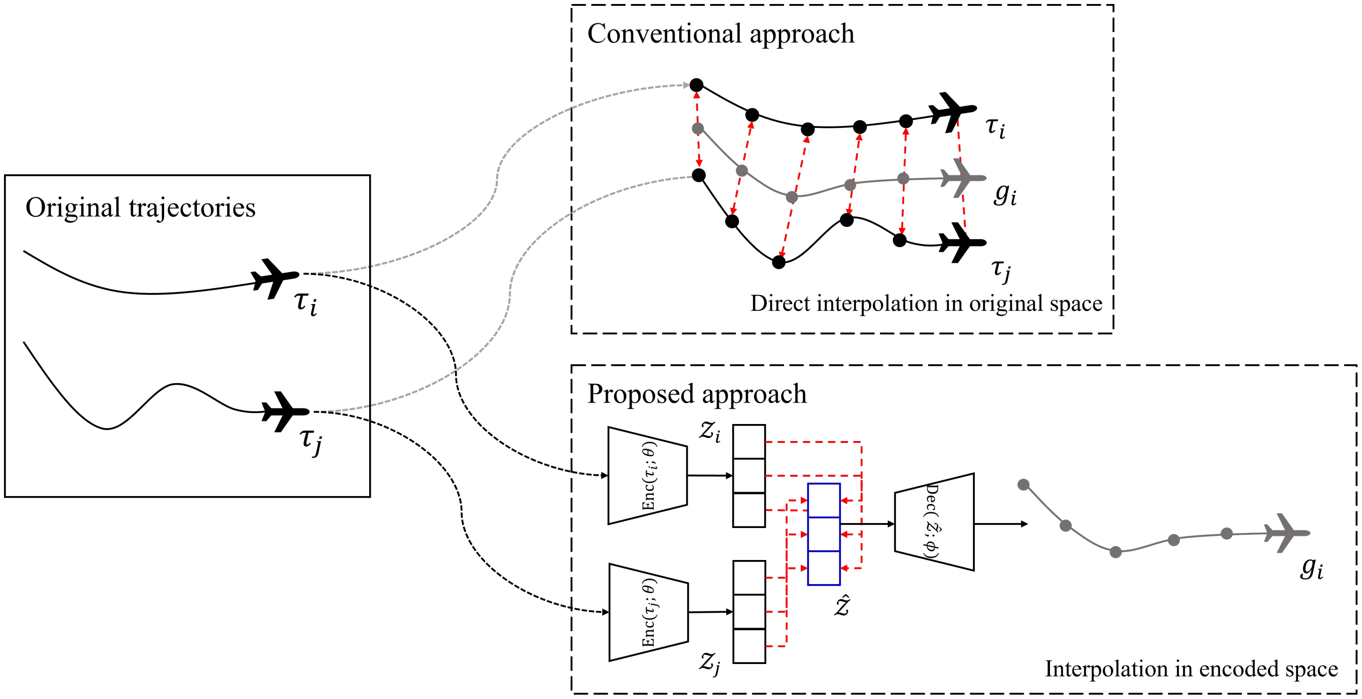

Original trajectories
τ_i
τ_j
Conventional approach
τ_i
g_i
τ_j
Direct interpolation in original space
Proposed approach
$\text{Enc}(\tau_i;\theta)$
$\mathcal{Z}_i$
$\text{Enc}(\tau_j;\theta)$
$\mathcal{Z}_j$
$\hat{\mathcal{Z}}$
$\text{Dec}(\hat{\mathcal{Z}};\phi)$
g_i
Interpolation in encoded space

Fig. 2.5: Comparison between conventional approach and proposed approach for trajectory reconstruction

2.4.1 Transformer

In this study, we adopt a Transformer architecture as an AE for trajecotry reconstruction [58], as depicted in Fig. 2.6. Unlike the standard AE, which typically consists of a Multi-Layer Perceptron Neural Network (MLP NN) and does not inherently process time-series data, and the Transformer can potentially be used for the same purpose (reconstruction) as an AE.

The Transformer architecture fundamentally differs from conventional sequential models like RNNs and LSTMs [59]. Unlike these models, which process input sequences sequentially, Transformers handle sequences in parallel. This characteristic addresses some limitations found in Sequence-to-Sequence (Seq2Seq) models [60]. In Seq2Seq models, compressing an entire input sequence into a fixed-size vector can lead to information loss, a problem that is particularly pronounced with longer sequences such as trajectories. In contrast, Transformer models utilize self-attention mechanisms to dynamically contextualize information, enabling the direct modeling of relationships across all parts of the input sequence. This approach allows for improved handling of long-range dependencies, free from the constraints of fixed-size context vectors.

The attention mechanism, the main part of Transformer, mimics human cognitive behavior, wherein selective focus is placed on certain parts of the information, rather than concurrently processing the entire spectrum of available information. The model attends to crucial points within a sequence, with this focus dynamically shifting depending on the particular context of each prediction. This is performed by assigning weights to each data point in the sequence, ensuring that the weights sum to one, with each weight indicating the

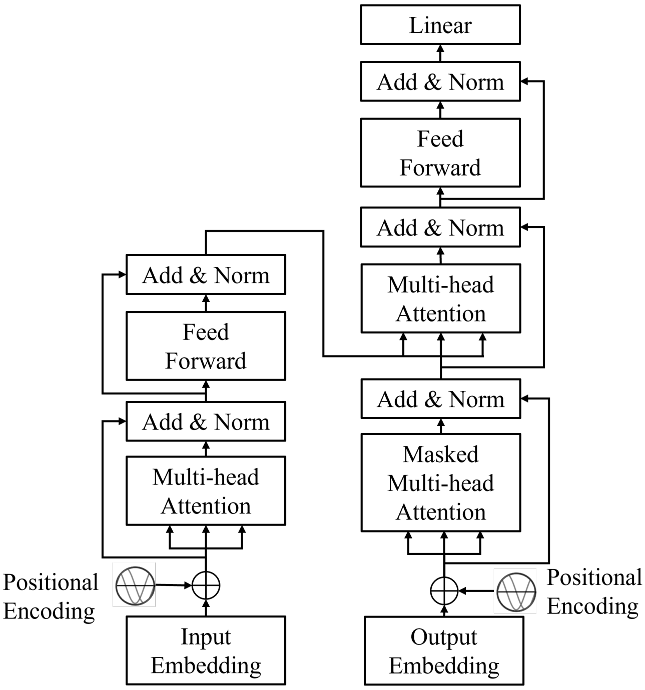


Fig. 2.6: Transformer architecture [58]

relative importance of the corresponding data point. The scaled dot-product attention can be characterized as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V \tag{8}$$

Here, the query denoted as Q serves as an information-seeking vector, aiming to identify relevant information among the keys represented by K. The keys K provide the answer (i.e., which specific data point are most relevant to the query Q) by the similarity measure from dot-product QK^T, which quantifies the similarity between the Q and K. Lastly, the values V are associated with the K and serve to represent the information corresponding to K. The attention mechanism can be thought of as a mapping function that takes the the query Q along with the key-value pairs K, V as its inputs. It processes this information to assign weights to the values V based on the similarity between the query Q and the keys K. These weighted values are then summed to produce the output, reflecting the importance of different elements in the input sequence. The output of dot-product is scaled by $\sqrt{d_K}$ because the magnitude of the dot-product can grow large, especially when the dimensionality d_K of K is high. This can lead to issues with very large or small gradients during the training.

Furthermore, multiple attention functions can be employed in parallel to learn diverse representations, with each attention head concentrating on a different aspect of the input data. The concept of multi-head attention, which consists of n number of heads, is defined as follows:

$$\text{MultiHeadAttention}(Q, K, V) = (head_1 \oplus head_2 \oplus \cdots \oplus head_n)W^O \tag{9}$$

where $head_h = \text{Attention}(QW_h^Q, KW_h^K, VW_h^V)$, and $W_h^Q, W_h^K, W_h^V, W^O, h = 1, \cdots, n$, are trainable weights. In the equation above, $\oplus$ denotes concatenation of vectors.

In this architecture, trajectories of considerable length can be reconstructed with high efficiency. However, simply replicating trajectories for dataset augmentation, as the AE might, could result in reproductions with limited variability, which is inappropriate for our purposes. To address this, we propose a novel approach consisting of three distinct stages, as illustrated in Fig. 2.7: (i) encoding all trajectories within a less frequent trajectory pattern, (ii) identifying the k-nearest neighbors using the encoded vectors and computing a weighted sum of these k vectors, and (iii) decoding the aggregated encoded vectors to reconstruct the trajectory.

By interpolating the encoding vectors, we generate trajectories similar to the original trajectories. The weights λ_i, randomly drawn and constrained to sum to one, are assigned to each encoding vector. The interpolation is performed as follows:

$$\begin{aligned} \hat{\mathcal{Z}} &= \sum_{i=1}^{k} \lambda_i \cdot \mathcal{Z}_i \\ \text{subject to} \quad & \sum_{i=1}^{k} \lambda_i = 1 \end{aligned} \tag{10}$$

The model specifications and training hyperparameters are detailed in Table 2.1.

Hyperparameter	**Detail**
Hidden dimension	32
Number of layers	3
Number of heads	2
Feedforward dimension	64
Optimizer	Adam
Number of epochs	200
Learning rate	$1e^{-3}$
Loss function	MSE (L2)
Dropout rate	0.1

Table 2.1: Trajectory reconstruction model specifications

2.4.2 Temporal Information Transfer

After augmenting our dataset with newly generated trajectories, we achieved a more balanced distribution among different trajectory patterns. However, this process introduced a significant challenge: the distortion of temporal information in generated trajectories. Due to the interpolation of original trajectories of varying lengths into a uniform length, which is the appropriate format for input

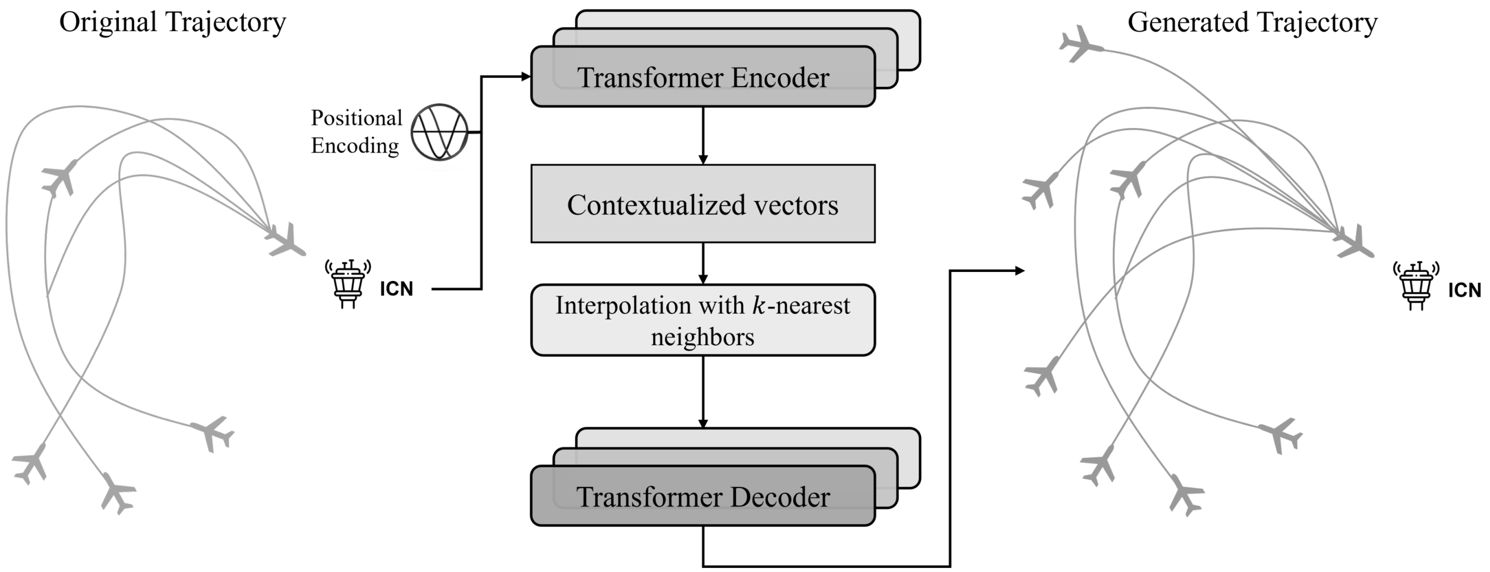


Fig. 2.7: Proposed approach for trajectory reconstruction

Algorithm 1 Temporal Information Transfer

```
Require: Trajectories $\mathcal{G} = \{g_1, g_2, \ldots, g_M\}$ and $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_N\}$
Ensure: Aligned trajectories with interpolated generated trajectories
Compute distance matrix $\mathcal{D}$ using DTW on $\mathcal{G}$ and $\mathcal{T}$
Identify closest point pairs $(g_i, \tau_j)$ using $\mathcal{D}$
for each closest point pair $(g_i, \tau_j)$ do
    Interpolate the number of data points in $g_i$ to match the number in $\tau_j$
end for
Construct the warping path $\mathcal{P}$ from the closest point pairs
return Aligned trajectories with interpolated generated trajectories
```

into GMM and Transformer, the time intervals in trajectories are distorted, being either stretched or compressed. This can lead to unrealistic time intervals between successive points in the generated trajectories. This irregular time intervals may affect the efficacy of trajectory prediction model, highlighting the need for maintaining realistic time intervals in the trajectory data. For instance, in predicting future trajectory sequences over a period of 2 minutes, the ambiguity in time intervals complicates determining the necessary number of time steps for accurate prediction.

To address this issue, we present the Temporal Information Transfer algorithm, as outlined in Algorithm 1. This algorithm utilizes Dynamic Time Warping (DTW) to measure distances between original trajectory and generated trajectory. DTW is chosen for its ability to assess similarity between two temporal sequences, irrespective of their differing lengths [61]. Consider two vectors, $x_1 = [x_1^{(1)}; \ldots ; x_1^{(m)}] \in \mathbb{R}^m$ and $x_2 = [x_2^{(1)}, \ldots , x_2^{(n)}] \in \mathbb{R}^n$, each representing a sequence of m and n time steps, respectively. The distance between i^{th} time step of x_1, and j^{th} time step of x_2 is denoted $d(i, j)$ for all $i \in \{1, \ldots , m\}, j \in \{1, \ldots , n\}$. By computing distances for each pair of indices i and j, a distance matrix is formed. Within this matrix, the warping path can be

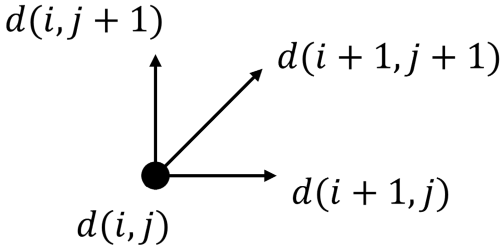


Fig. 2.8: Three possible direction for each element

identified. The warping path $\mathcal{P}$ is a sequence of elements within the matrix that traces the most effective alignment between the two sequences. The identification of the warping path $\mathcal{P}$ is determined through the following equation:

$$d(i,j) = \|x_1^i - x_2^j\|_2 + \min \begin{cases} d(i+1,j), \\ d(i,j+1), \\ d(i+1,j+1). \end{cases} \tag{11}$$

In the above equation, $d(i,j)$ represents the cumulative distance at each element (i,j) in the matrix, computed as a sum of two components: the Euclidean distance between the i^{th} element of x_1 and the j^{th} element of x_2, denoted as $\|x_1^i - x_2^j\|_2$, and the minimum of three possible preceding steps in the matrix: $d(i+1,j), d(i,j+1), d(i+1,j+1)$, which is illustrated in Fig. 2.8.

We then construct a distance matrix $\mathcal{D}$, where each row corresponds to an index from the set of original trajectories $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_N\}$, and each column corresponds to an index from the set of generated trajectories $\mathcal{G} = \{g_1, g_2, \ldots, g_M\}$, and the matrix values denote the distance between these trajectories, as illustrated in Fig. 2.9. The algorithm identifies pairs of trajectories with the shortest distances. For each identified pair, the generated trajectory is

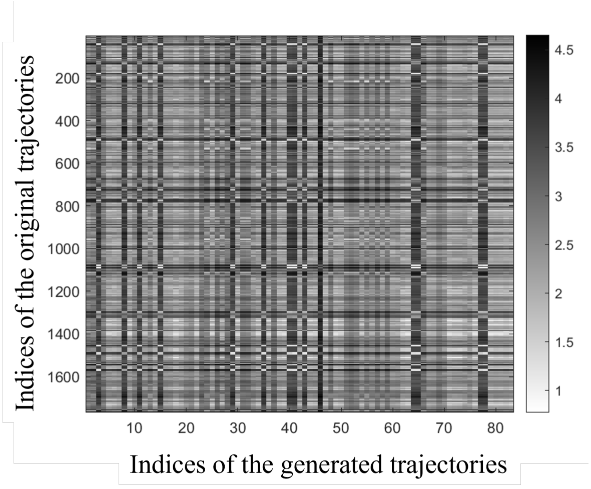


Fig. 2.9: An example of distance matrix by DTW

re-interpolated to match the number of data points in the original trajectory, thereby aligning the time interval of the generated trajectory with that of the original. This process is based on the assumption that if two trajectories are very similar (indicated by a short distance), their total flight times should also be very similar.

Figure 2.10 presents a comparison across four dimensions—north, east, up, and speed—between two types of trajectories: one re-interpolated and another un-interpolated. In the first three figures, a significant gap is observed between the two trajectories, despite both being plotted with the same number of data points (30). The un-interpolated trajectory exhibits unrealistic intervals between points, indicating abrupt speed changes during the arrival stage. This is inconsistent with typical aircraft behavior and could adversely influence learning since the original trajectory does not exhibit such speed variations. In the last figure, the un-interpolated trajectory demonstrates a significantly higher speed compared to the re-interpolated trajectory, again diverging from expected aircraft

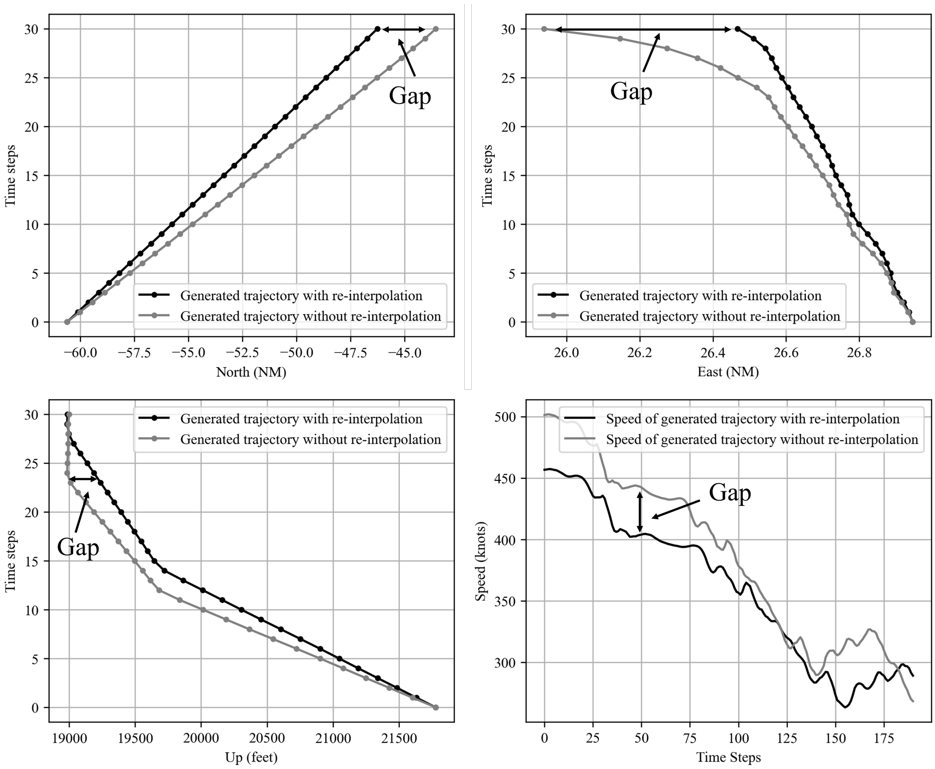


Fig. 2.10: Comparison of re-interpolated and un-interpolated trajectories

behavior during the arrival stage. Note that the re-interpolated trajectory is adjusted based on the number of data points from the trajectory with the shortest distances using a temporal information transfer algorithm, enabling it to more accurately reflect the speed profile of the original trajectory.

2.5 Trajectory Prediction

In this section, we explore the basics of LSTM and the structure of a trajectory prediction model using Sequence to Sequence architecture.

2.5.1 Long Short-Term Memory

Long Short-Term Memory (LSTM) networks have since been a cornerstone of the field of deep learning [59]. LSTMs are a subset of Recurrent Neural Networks (RNNs) [62], a class of neural networks that possess a form of memory allowing them to process sequential data. The key innovation of LSTM networks is their capacity to retain and selectively recall information over lengthy sequences, overcoming the vanishing gradient issue common in traditional RNNs. An LSTM network operates through internal states that store information from previous inputs, managed by a complex array of gating units that regulate memory flow. The structure of these gates within an LSTM cell can be seen in Fig. 2.11, and can be formulated as follows:

$$\begin{aligned}
\Gamma_{f_t} &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
\Gamma_{i_t} &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\
C_t &= \Gamma_{f_t} \odot C_{t-1} + \Gamma_{i_t} \odot \tilde{C}_t \\
\Gamma_{o_t} &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
h_t &= \Gamma_{o_t} \odot \tanh(C_t)
\end{aligned} \tag{12}$$

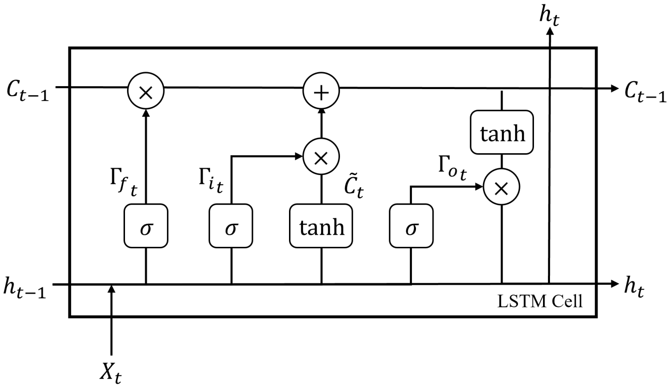


Fig. 2.11: The structrue of LSTM cell

where x_t denotes the input at time step t, h_{t-1} and C_{t-1} represent the hidden and cell states from the previous time step, and Γ_{f_t}, Γ_{i_t}, Γ_{o_t} denote the forget, input, and output gates at time t, respectively. $\tilde{C}_t$ signifies the candidate cell state. Element-wise multiplication is denoted by $\odot$, while σ and $\tanh$ refer to the sigmoid and hyperbolic tangent activation functions. The terms W_f, W_i, W_C, and W_o represent the weight matrices that the network learns during training, and b_f, b_i, b_C, and b_o denote the bias vectors, which are consistent and applied at each time step.

2.5.2 Sequence-to-Sequence Architecture

To enhance the prediction of future sequences based on given input sequences,we utilize a Sequence-to-Sequence (Seq2Seq) model [60], incorporating LSTM in both encoder and decoder, as shown in Fig. 2.12. Note that the Transformer architecture can also be employed for trajectory prediction. However, for short to medium-length sequences, particularly in the context of trajectory prediction, our findings indicate that LSTM tends to outperform the Transformer. Nevertheless,

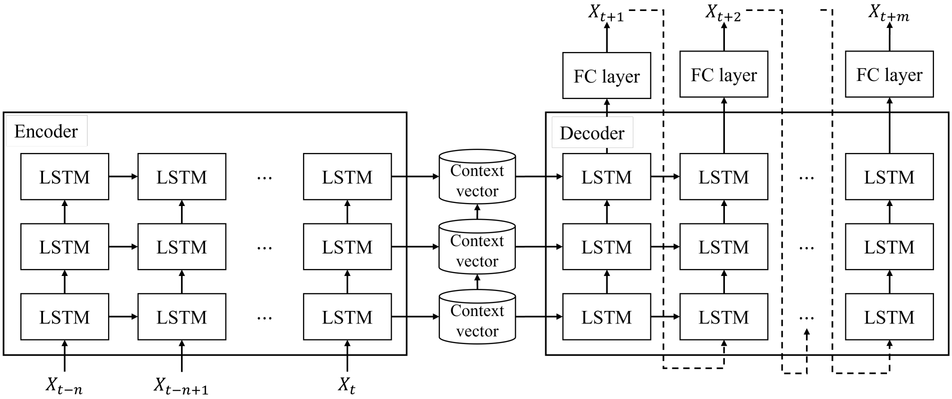


Fig. 2.12: Seq2Seq architecture for trajectory prediction

for trajectory prediction involving longer sequences, the Transformer architecture also should be explored carefully. Originally designed for machine translation, the Seq2Seq is adept at handling tasks involving sequential input and output. Here, the Encoder processes and compresses the input sequence into a context vector $\mathcal{C}$. The Decoder then processes this context vector to generate predictions, using its previous output as the input for the next time step, a process known as the autoregressive configuration. The Seq2Seq model can be represented by the following two main equations:

$$\begin{aligned} \mathcal{C} &= \text{Enc}(X; \theta) \\ \hat{X}_{t+i} &= \text{Dec}(\mathcal{C}, \hat{X}_{t+i-1}; \phi) \end{aligned} \tag{13}$$

where X represents the input sequence, $\hat{X}_{t+i-1}$ denotes the predicted value at time step $t+i-1$, and $\hat{X}_{t+i}$ is the predicted value at time step $t+i$. θ represents the parameters of the encoder, and ϕ denotes the parameters of the decoder.

To further enhance predictions, our architecture incorporates an attention mechanism [63]. This mechanism produces an attention context vector, which

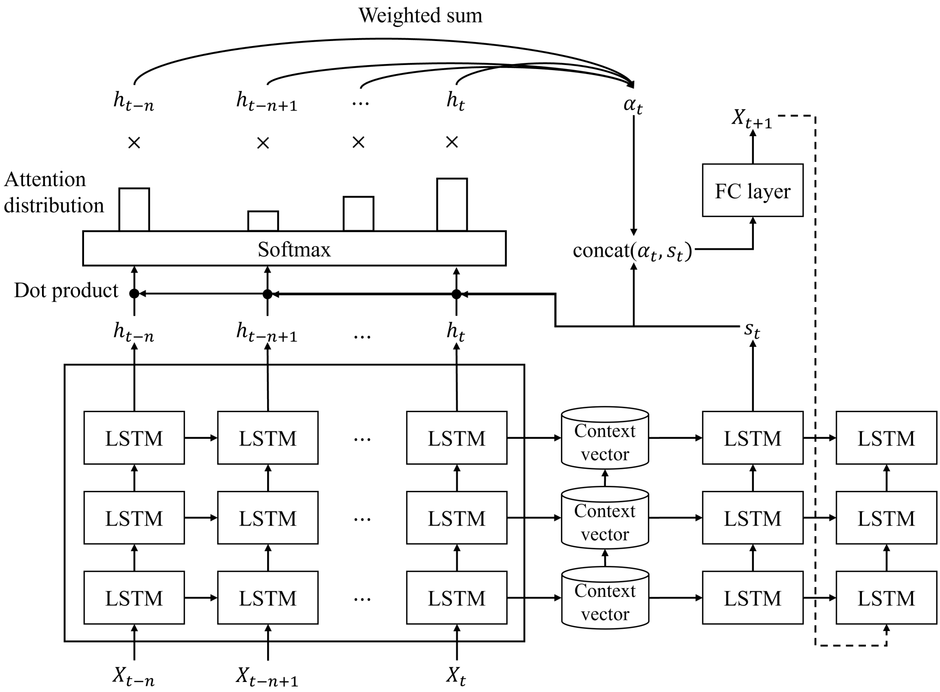


Fig. 2.13: Attention mechanism for trajectory prediction

can be utilized in two ways: (i) by concatenating it with the decoder's hidden state, or (ii) by replacing the decoder's hidden state with it. Our study employs concatenation, as shown in Fig. 2.13, to preserve sequential context from the decoder's hidden state, which might be lost if relying solely on the attention context vector. This approach concatenates two key types of information: specific portions of the input sequence at the current stage from the attention context, and the cumulative output sequence from the decoder's hidden state. This process enhances the model's ability to make informed predictions for subsequent outputs. Note that although the attention mechanism equations in both the Transformer and Seq2Seq models are identical, their applications differ. In the Seq2Seq, the query Q is represented by the decoder's hidden state, while the keys K and values V correspond to the encoder's hidden state. This contrasts with the self-attention mechanism in the Transformer model, where Q, K, and V are all from same sequence. In other words, the attention mechanism in the Seq2Seq model is used to identify which data points in the input sequence are most relevant when making a prediction. Conversely, in the Transformer, the attention mechanism serves to contextualize each data point within the input sequence in relation to all others. The model specifications and training hyperparameters are detailed in Table 2.2.

Hyperparameter	**Detail**
Hidden dimension	256
Number of layers	3
Number of FC layers	3
FC activation function	ReLu
Teacher forcing ratio	0.5
Optimizer	Adam
Number of epochs	200
Learning rate	$1e^{-3}$
Loss function	MSE (L2)
Dropout rate	0.1

Table 2.2: Trajectory prediction model specifications

3 Numerical Examples

In this section, we evaluate our proposed approach and provide the results for trajectory reconstruction and prediction in the vectored area navigation airspace of Incheon International Airport. Both trajectory reconstruction and prediction models were trained on a desktop featuring an Intel Core i7-12700F CPU (20 cores), 32 GB RAM, and an NVIDIA GeForce RTX 3060 GPU. Pytorch was used for model development [64], and mixed precision training was applied to improve training efficiency [65].

3.1 Airspace of Interest

In this study, we focus on arrival aircraft trajectories in vectored area navigation airspace of Incheon International Airport, which is within 70 NM radius from the airport reference point. The arrival aircraft trajectory usually enters this airspace via one of four designated entry fixes: REBIT, OLMEN, GUKDO, and KARBU. Aircraft entering through REBIT usually originate from China and Europe, while those passing OLMEN often come from Southeast Asia and Jeju, South Korea. Furthermore, aircraft entering through GUKDO are typically flying from Japan and Oceania, and those passing through KARBU are mostly from the Americas. Fig. 3.1 shows the vectored area navigation airspace of Incheon International Airport and STARs under RWY 15/33 and RWY 16/34 configurations.

In the development of the trajectory reconstruction model, the trajectory dataset within this airspace was segmented into training, validation, and test dataset in an 8:1:1 ratio. Similarly, for the trajectory prediction model, the

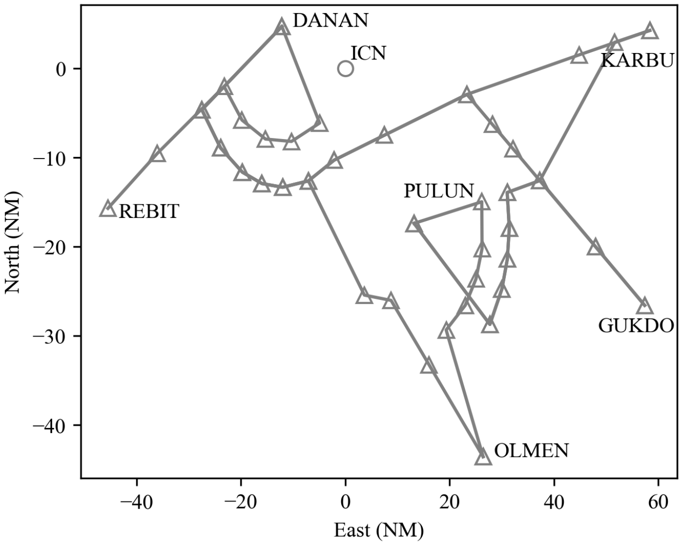


Fig. 3.1: Vectored area navigation airspace of Incheon International Airport

Dataset	**Training**	**Validation**	**Test**
Trajectory Reconstruction	19,910	4,977	4,977
Trajectory Prediction	38,855,156	485,645	485,645

Table 3.1: Trajectory dataset segmentation

trajectory dataset was partitioned following the same 8:1:1 ratio. The training dataset is used to train the model, enabling it to learn the patterns within the dataset. The validation dataset provides an unbiased evaluation of the model's fit during the training phase, facilitating hyperparameter tuning. Finally, the test dataset is employed to assess the model's performance after training. Note that the validation dataset is incorporated into the training dataset after hyperparameter tuning is complete. Table 3.1 displays the resulting number of samples in each dataset. The counts for each dataset in the two models vary due to the application of a sliding window technique in the trajectory prediction model, which divides a trajectory into multiple subsequences.

3.2 Trajectory Reconstruction

In Section 2.3, we identify a total of 11 infrequent trajectory patterns. Trajectory reconstruction is performed to reconstruct and augment these trajectory patterns. During training, all trajectories from these infrequent patterns are input into a trajectory reconstruction model. For reconstruction, trajectories from each pattern are individually processed by the model, ensuring that results are not influenced by trajectories from other patterns. The top part of Fig. 3.2 illustrates the original (real) trajectories on the left, with a histogram on the right indicating the count of trajectories within each pattern. This histogram highlights a considerable variance in the number of trajectories across different patterns. The bottom part of Fig. 3.2 displays the generated trajectories on the left and the associated histogram on the right, depicting the number of trajectories for each pattern following trajectory reconstruction, which demonstrates a more balanced distribution.

To evaluate if the trajectory reconstruction model accurately captures the realistic physical properties of arriving aircraft, we analyze the horizontal and vertical speeds of both generated and actual trajectories across each trajectory pattern. Note that the trajectory reconstruction is performed for only less frequent trajectory patterns. The speed distributions, illustrated in Figures 3.3 and 3.4, serve as indicators of the model's fidelity in replicating the actual dynamics of aircraft movements. These figures indicate that the model effectively learns and replicates the realistic physical characteristics of arriving aircraft, even though the generated trajectories are derived through interpolation.

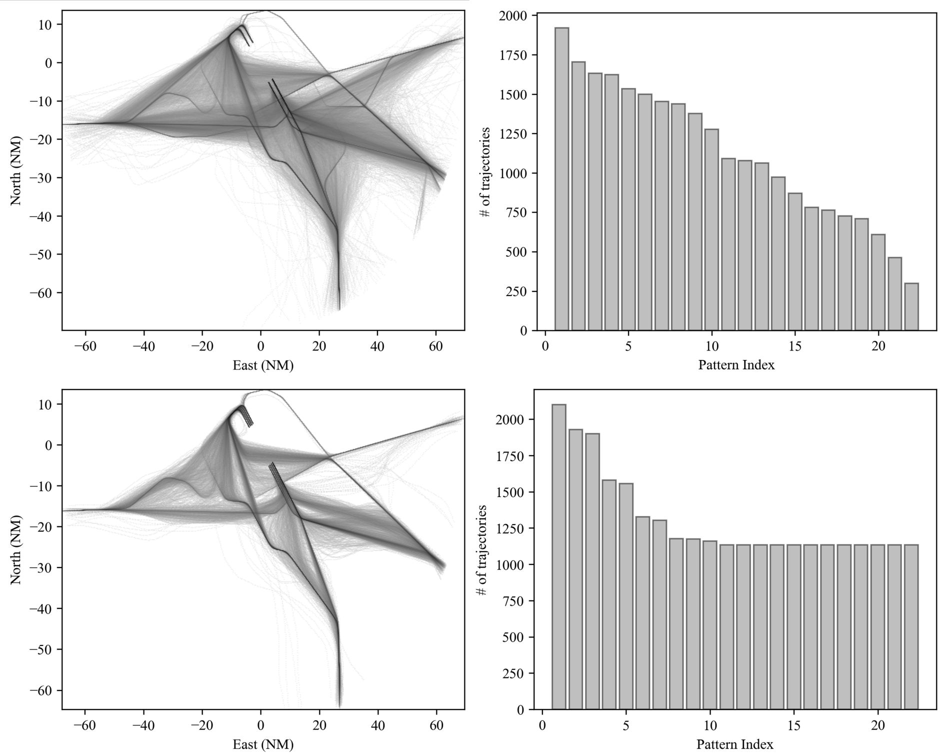


Fig. 3.2: Real (top) and **generated (bottom)** trajectories at Incheon International Airport

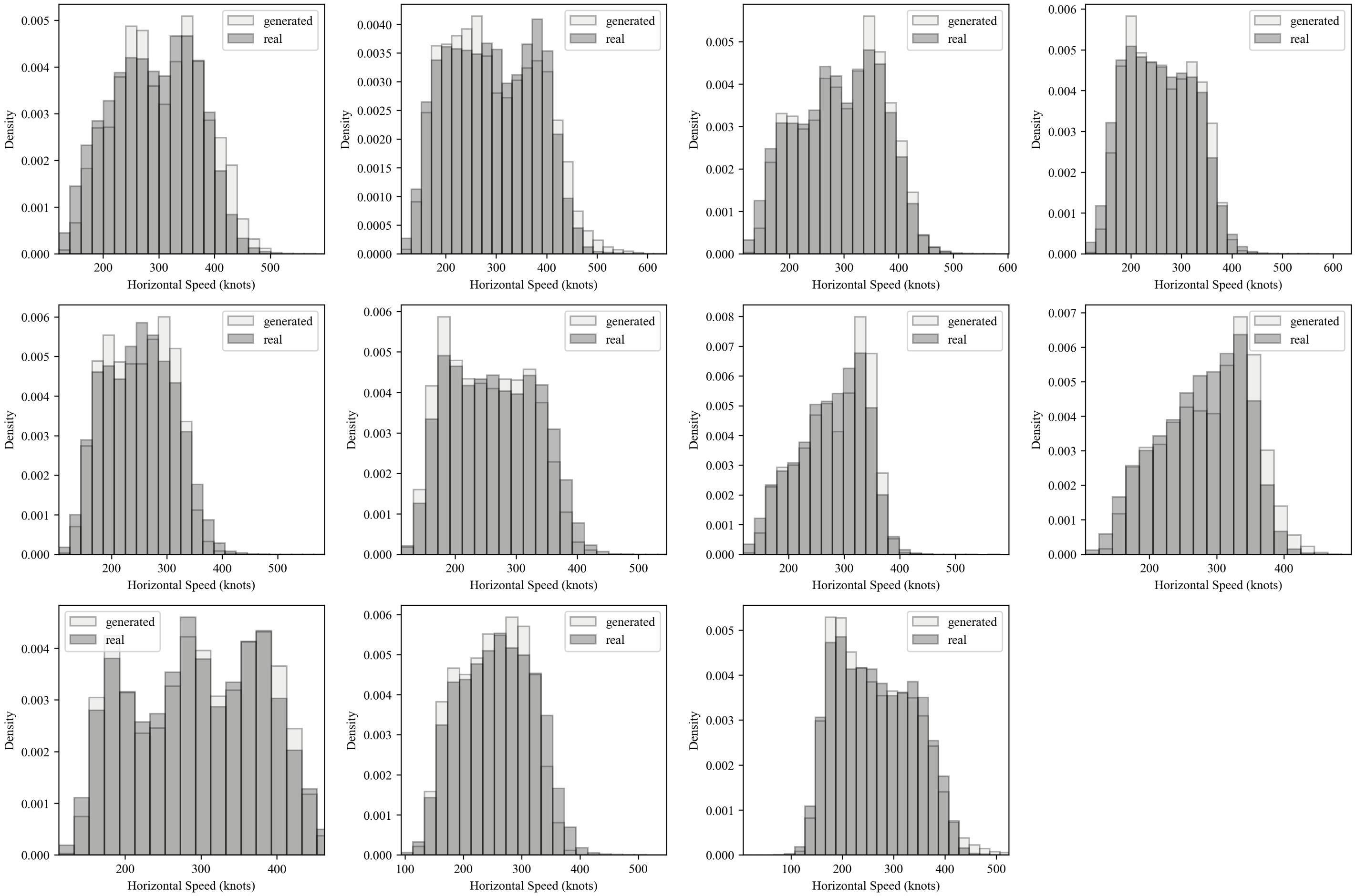


Fig. 3.3: Horizontal speed distributions of **generated** and **real** trajectories

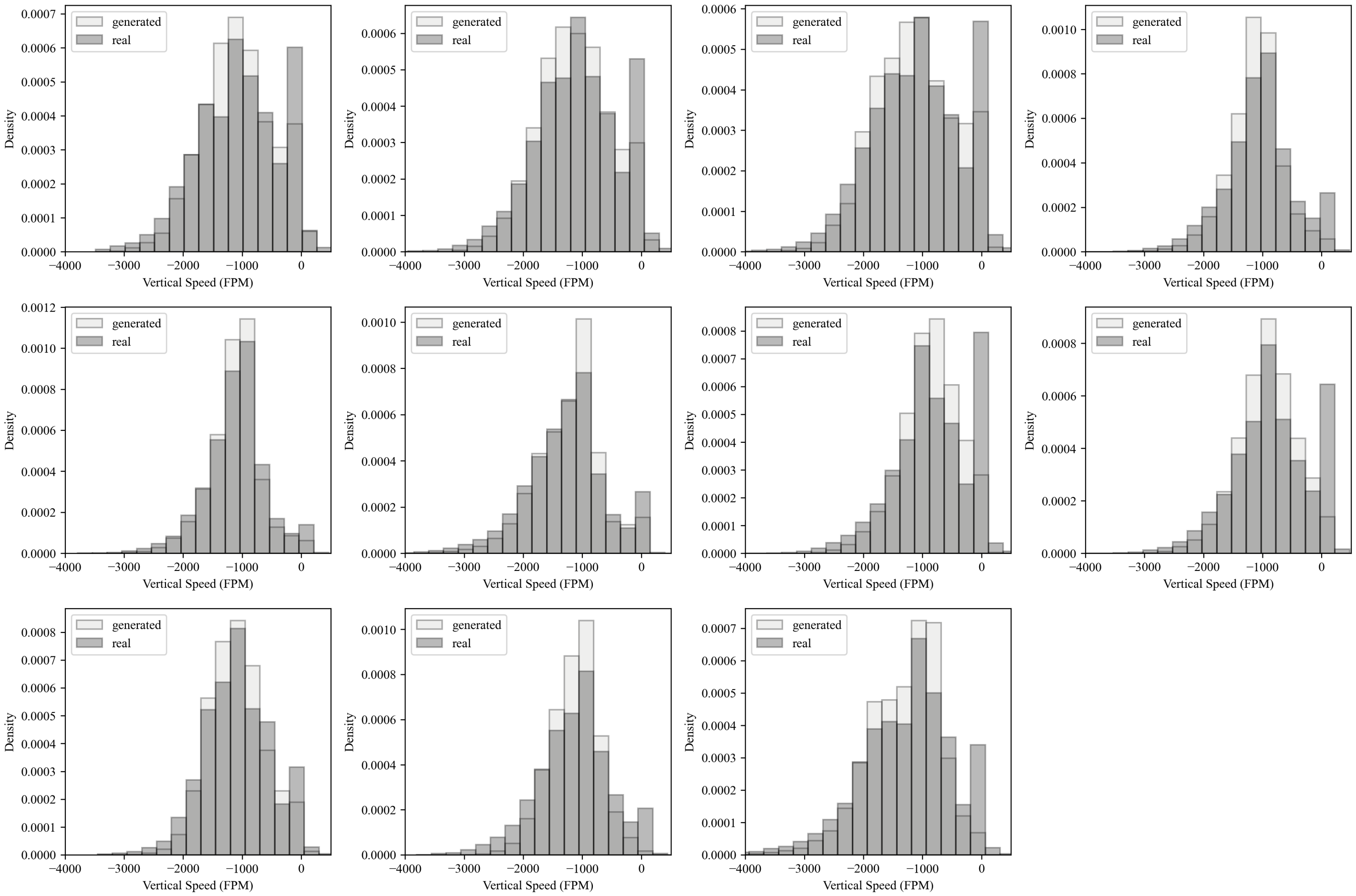


Fig. 3.4: Vertical speed distributions of **generated** and **real** trajectories

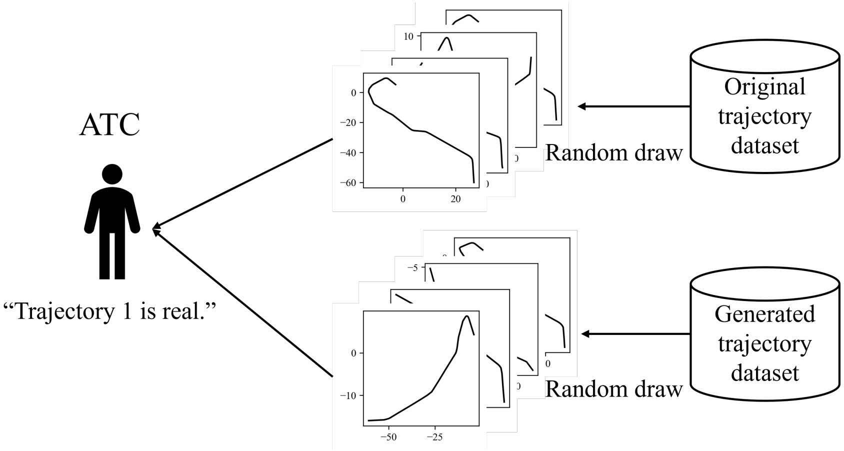


Fig. 3.5: Example of ATC Turing Test

3.2.1 ATC Turing Test

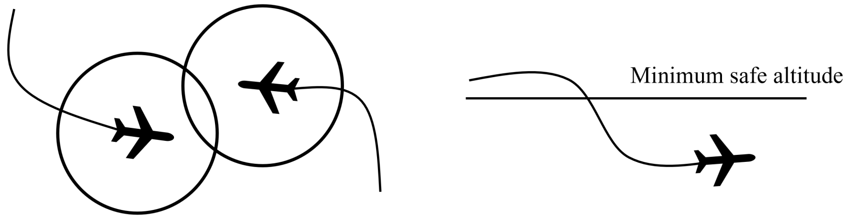
To evaluate the realism of the generated trajectories, we conducted an ATC Turing Test, a commonly used method for validating the realism of generated trajectories [33, 66]. In this test, subjects are asked to differentiate between trajectories generated by the reconstruction model and actual trajectories, as illustrated in Fig. 3.5. We selected 10 random samples from both the generated and real trajectory datasets. Each ATC was presented with 10 pairs of these trajectories. Each pair included a lateral view (east, north) and an altitude profile. We calculated the accuracy of their identifications to determine how well the trajectories could be differentiated.

The ATC Turing Test was conducted with four ATCs who have experience managing terminal airspace at Incheon International Airport. As shown in Table 3.2, participants correctly identified model-generated trajectories only 40% of the time, on average. This low rate of correct identification suggests that the model-generated trajectories were often perceived as real by these experienced

Subject	**Accuracy**
ATC 1	20%
ATC 2	40%
ATC 3	60%
ATC 4	40%
Average	40%

Table 3.2: ATC Turing Test results

ATCs. Such results suggest that our trajectory reconstruction model successfully mimics realistic aircraft behavior, creating trajectories that closely resemble actual trajectory patterns.



Conflict Alert (CA)   Minimum Safe Altitude Warning (MSAW)

Fig. 3.6: Example of CA and MSAW

3.3 Trajectory Prediction

Trajectory prediction is conducted to assess and compare the performance of the proposed approach with the conventional approach. For this, two models are trained: one using an augmented training dataset and the other using the original dataset. To maintain fairness and consistency in evaluation, both models are trained under identical conditions, with the details of this training environment outlined in Table 2.2.

In the paper [67], median response times for ATCs are recorded as 88 seconds for Conflict Alerts (CA) and 38 seconds for Minimum Safe Altitude Warnings (MSAW). Fig. 3.6 illustrates the examples of CA and MSAW. These response times are measured from the activation of an alert to the issuance of air traffic control instructions. In light of this, we have set the trajectory prediction horizon to 2 minutes. This prediction horizon, exceeding the longest median response time, provides ATCs with sufficient time for effective decision-making. Specifically, the advantage of this prediction horizon include: (i) providing a buffer to account for potential delays in decision-making processes, and (ii) reducing the need for ATCs to continuously monitor the situation, thus enabling earlier and potentially

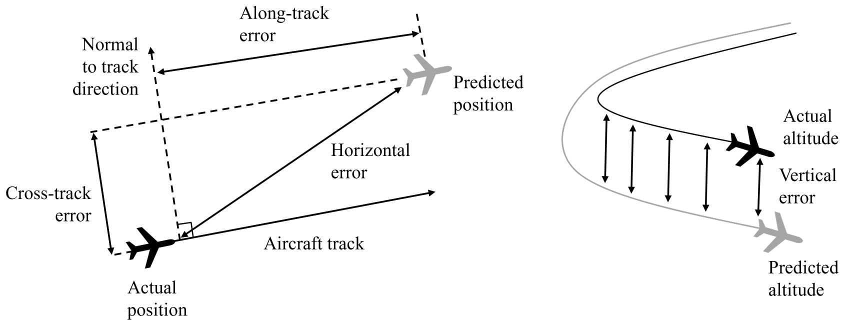


Fig. 3.7: Evaluation metrics for trajectory prediction

more efficient interventions. In our evaluation, predictions span a horizon of 30 timesteps, equivalent to 120 seconds, with each timestep having a duration of $\Delta t = 4$ seconds. The extension to a longer prediction horizon is left for future work.

3.3.1 Evaluation Metrics

In order to evaluate the performance of our prediction model, we utilized four evaluation metrics described in [68, 69, 70, 71]. The Horizontal Error (HE) measures the discrepancy between the actual and predicted positions of an aircraft in the horizontal plane. The Along-Track Error (ATE) gauges the deviation along the actual flight path, while the Cross-Track Error (CTE) assesses the perpendicular deviation from this path. The Vertical Error (VE) denotes the difference in altitude, representing vertical deviations. These four metrics are illustrated in Fig. 3.7

3.3.2 Prediction Results

In order to demonstrate the effectiveness of our approach, we present three illustrative examples of prediction results. Figure 3.8 displays these for three arrival trajectories, with the top-down view on the left and side views on the right. Observations from Figure 3.8 show that while the baseline model generates predictions aligning with the overall trend, our proposed model yields more accurate predictions, closely matching the actual ground truth trajectory.

We also observe a reduction in trajectory fluctuations, resulting in smoother predicted trajectories. This enhancement is due to the model's improved generalization ability, achieved by training with a more varied set of examples compared to the original dataset. This diversity in training data contributes significantly to the model's robust performance during evaluation.

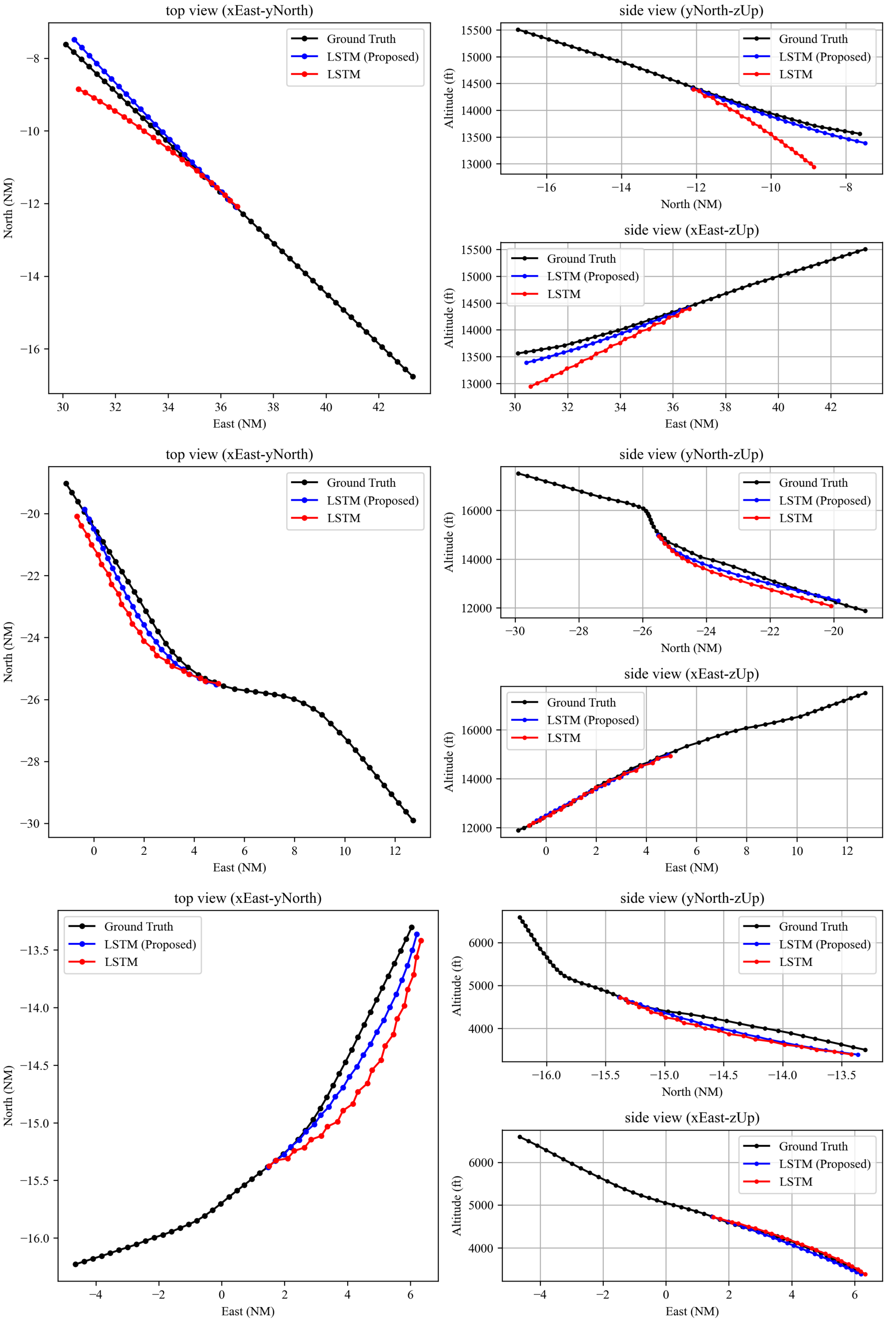


Fig. 3.8: Illustrative prediction results of two trajectory prediction models. Top view (left column) and side views from north and east (right column)

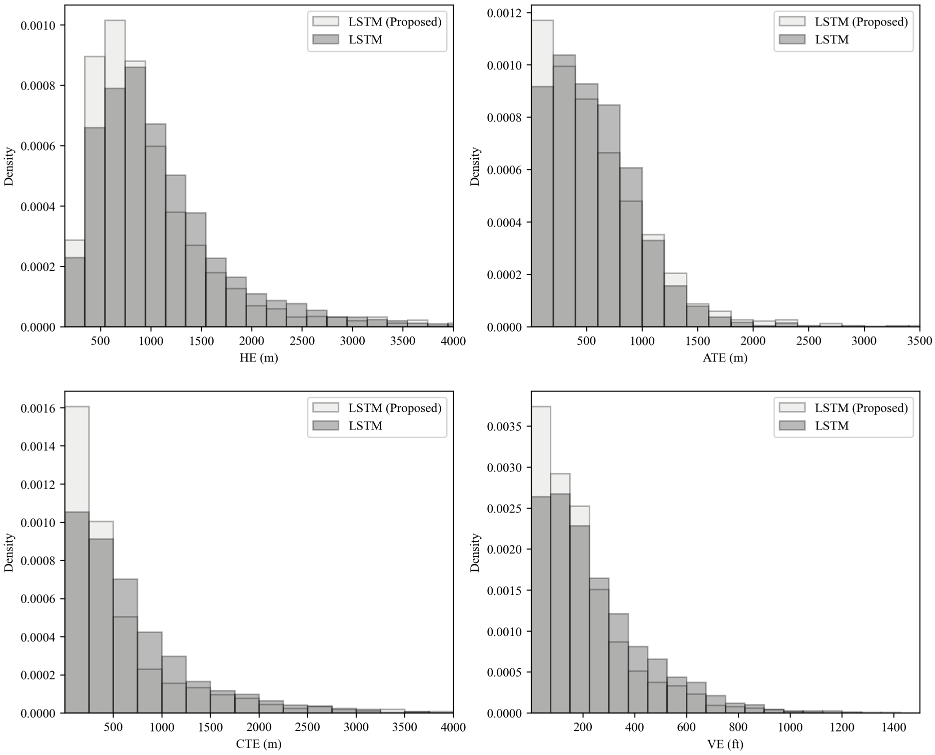


Fig. 3.9: Histogram of HE, ATE, CTE and VE of two trajectory prediction models

To evaluate the performance of the trajectory prediction model, we computed four metrics: HE, ATE, CTE, VE, as defined in Section 3.3.1. These metrics were calculated based on the predicted trajectories and the corresponding ground truth trajectories. For clarity in comparison, we used the absolute values of these error metrics, even though some can have negative values. Fig. 3.9 displays histograms for these four metrics. In these histograms, the proposed model shows a higher concentration of values near zero compared to the baseline model, indicating better accuracy. More importantly, there is a reduction in error variance, highlighting the model's robust prediction capability. This suggests consistent performance by our model, irrespective of the trajectory pattern's frequency.

4 Conclusion

4.1 Concluding Remarks

Motivated by the need for the development of an accurate trajectory prediction model in terminal airspace, particularly within vectored area navigation airspace, a novel framework has been developed to address the inherent imbalances in trajectory dataset. This framework employs the Transformer architecture for trajectory reconstruction, augmenting the training dataset with reconstructions of infrequent trajectory patterns. It also utilizes an LSTM network for trajectory prediction. The proposed framework effectively generates trajectories that are similar to the original ones while introducing variation. Its effectiveness has been validated using real trajectory data from Incheon International Airport. When compared to a baseline model trained on the original dataset, the proposed framework demonstrates superior performance across four evaluation metrics.

4.2 Possible Applications

We believe that this framework could be extended to the development of other decision support tools for ATCs. For example, this framework can be easily modified for developing Estimated Time of Arrival (ETA) prediction models. Moreover, the trajectory reconstruction model holds significant potential in developing a Collision Risk Model (CRM). This is particularly crucial given the infrequency of collisions; accurate collision risk modeling requires a substantial amount of trajectory data, which the trajectory reconstruction model can easily provide. Additionally, the trajectory reconstruction model could be repurposed for simulating new flight procedures with minor adjustments to the data format.

4.3 Future Work

For future work, it would be beneficial to investigate additional techniques for trajectory pattern identification, incorporating domain knowledge to refine this process. Exploring alternative generative models like Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) could lead to advancements in trajectory reconstruction. Furthermore, modifying the loss function in the training of trajectory prediction models to more directly reflect aircraft movements could be beneficial.

Appendix

Further details of the trajectory reconstruction model and the trajectory prediction model are provided in Table A.1 and A.2. The shape is represented as (batch size, sequence length, dimension), where ℓ denotes the timestep in the decoder. Both models employ an autoregressive configuration in the decoder.

Layer	**Input Shape**	**Output Shape**
Input Embedding	(16, 250, 3)	(16, 250, 32)
Positional Encoding	-	(16, 250, 32)
Encoder		
Self-Attention	(16, 250, 32)	(16, 250, 32)
Residual Connection	(16, 250, 32)	(16, 250, 32)
Layer Normalization	(16, 250, 32)	(16, 250, 32)
Feedforward 1	(16, 250, 32)	(16, 250, 64)
ReLu Activation function	(16, 250, 64)	(16, 250, 64)
Feedforward 2	(16, 250, 64)	(16, 250, 32)
Residual Connection	(16, 250, 32)	(16, 250, 32)
Layer Normalization	(16, 250, 32)	(16, 250, 32)
Decoder		
Masked Self-Attention	(16, ℓ, 32)	(16, ℓ, 32)
Residual Connection	(16, ℓ, 32)	(16, ℓ, 32)
Layer Normalization	(16, ℓ, 32)	(16, ℓ, 32)
Cross-Attention	(16, 250, 32), (16, ℓ, 32)	(16, ℓ, 32)
Residual Connection	(16, ℓ, 32)	(16, ℓ, 32)
Layer Normalization	(16, ℓ, 32)	(16, ℓ, 32)
Feedforward 1	(16, ℓ, 32)	(16, ℓ, 64)
ReLu Activation function	(16, ℓ, 64)	(16, ℓ, 64)
Feedforward 2	(16, ℓ, 64)	(16, ℓ, 32)
Residual Connection	(16, ℓ, 32)	(16, ℓ, 32)
Layer Normalization	(16, ℓ, 32)	(16, ℓ, 32)
Output Embedding	(16, ℓ, 32)	(16, ℓ, 3)

Table A.1: Structure of Transformer

Layer	**Input Shape**	**Output Shape**
Encoder		
LSTM Layer 1	(128, 30, 3)	(128, 30, 256)
LSTM Layer 2	(128, 30, 256)	(128, 30, 256)
LSTM Layer 3	(128, 30, 256)	(128, 30, 256)
Decoder		
LSTM Layer 1	(128, ℓ, 3)	(128, ℓ, 256)
LSTM Layer 2	(128, ℓ, 256)	(128, ℓ, 256)
LSTM Layer 3	(128, ℓ, 256)	(128, ℓ, 256)
Cross-Attention	(128, 30, 256), (128, ℓ, 256)	(128, ℓ, 512)
Feedforward 1	(128, ℓ, 512)	(128, ℓ, 256)
ReLu Activation function 1	(128, ℓ, 256)	(128, ℓ, 256)
Feedforward 2	(128, ℓ, 256)	(128, ℓ, 128)
ReLu Activation function 2	(128, ℓ, 128)	(128, ℓ, 128)
Feedforward 3	(128, ℓ, 128)	(128, ℓ, 3)

Table A.2: Structure of Seq2Seq LSTM

Fig. A.1 and Fig. A.2 illustrate the original and generated trajectories for a less frequent trajectory pattern, respectively.

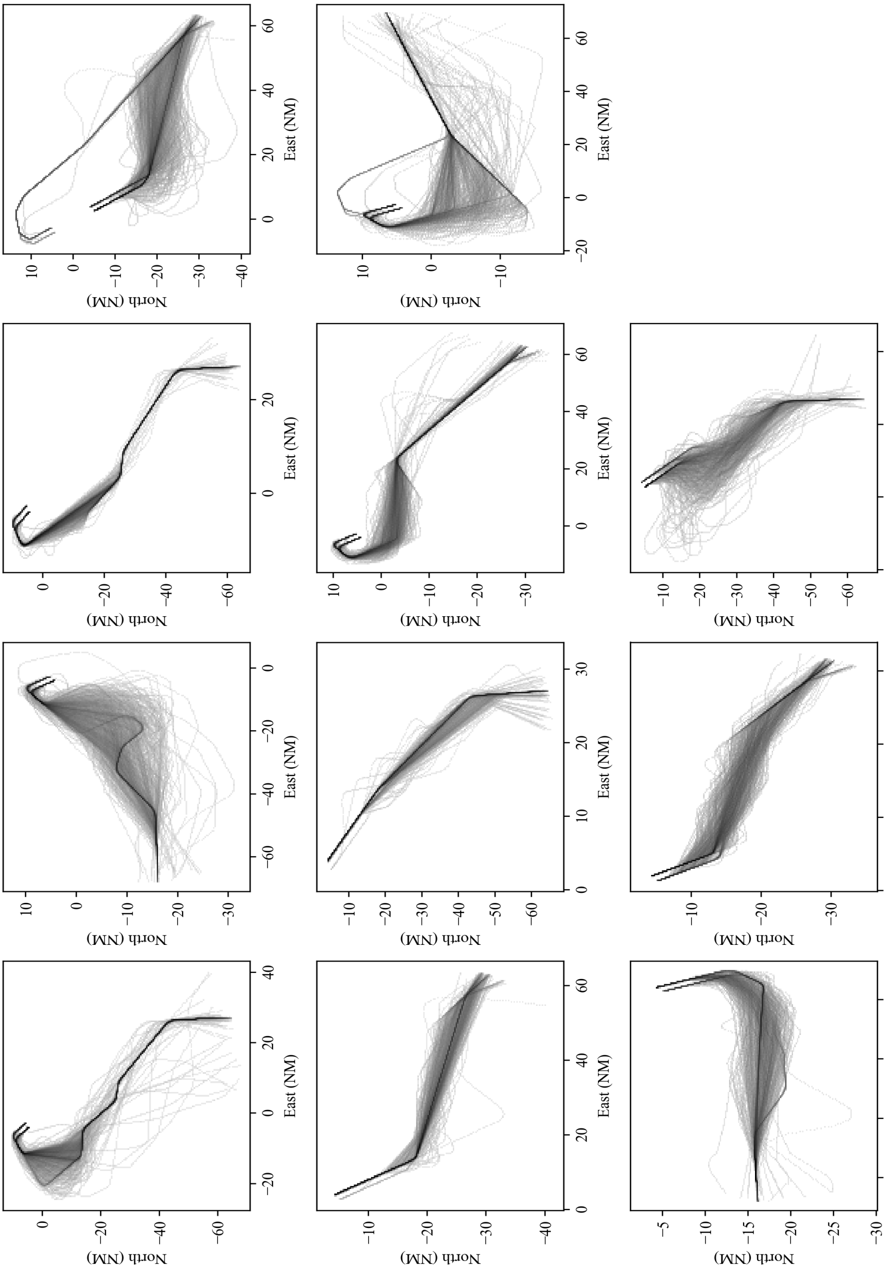


Fig. A.1: Original trajectories in less frequent trajectory pattern

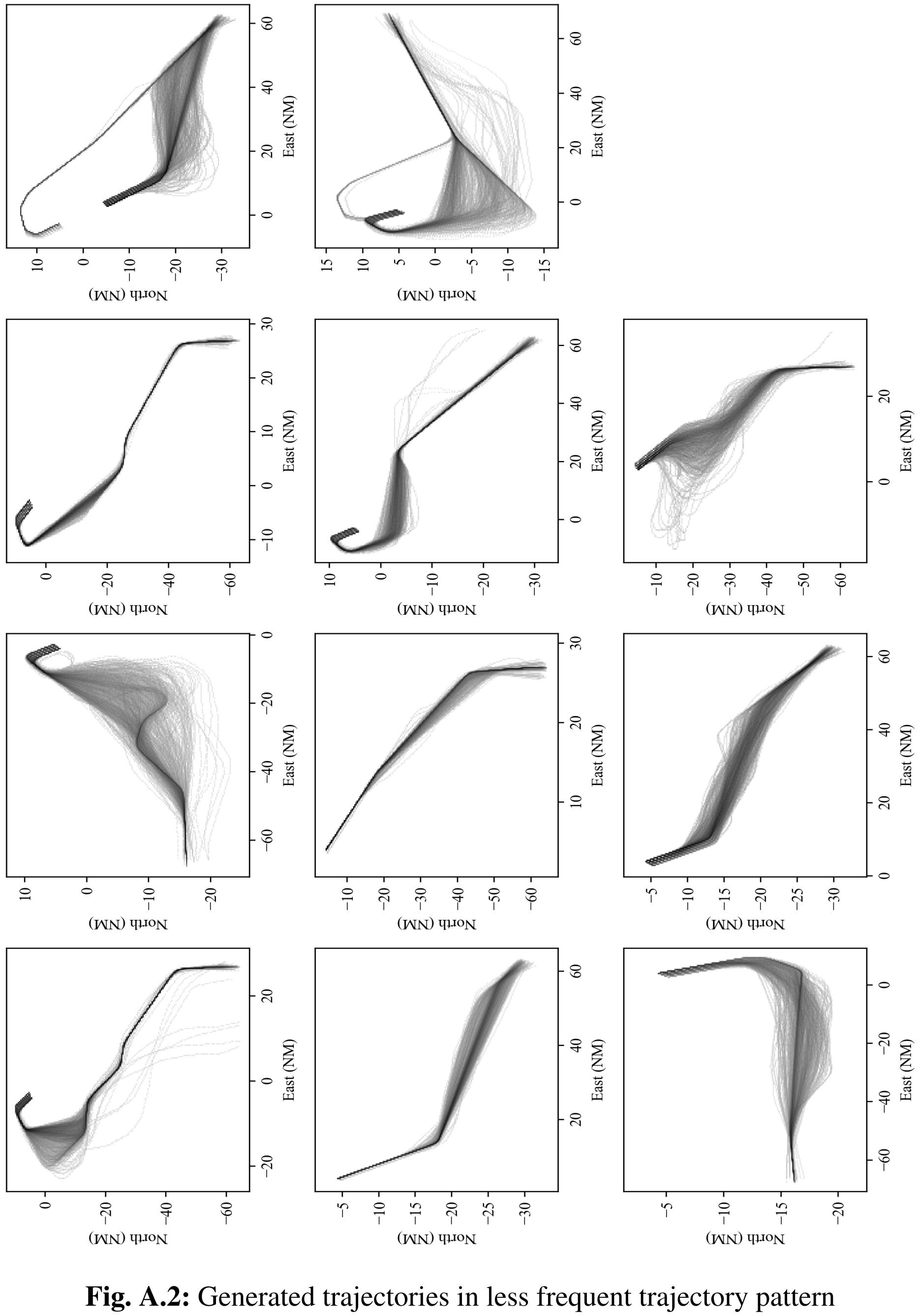
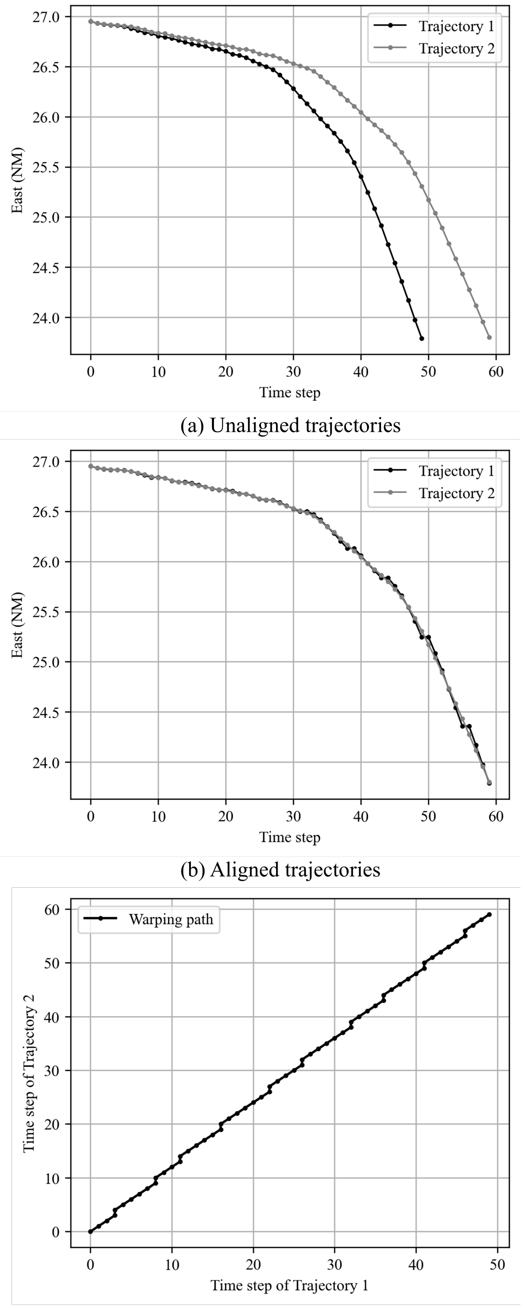


Fig. A.2: Generated trajectories in less frequent trajectory pattern

Fig. A.3 illustrates the application of dynamic time warping to trajectories.



(a) Unaligned trajectories

(b) Aligned trajectories

(c) Warping path

Fig. A.3: Example of dynamic time warping

References

[1] Planning, J. et al. (2007). Concept of operations for the next generation air transportation system. *Technical report*. Citeseer.

[2] Undertaking, S. J. (2007). Sesar milestone deliverable d3-the atm target concept.

[3] Swenson, H., Barhydt, R., & Landis, M. (2006). *Next generation air transportation system (ngats) air traffic management (atm)-airspace project* (tech. rep.). Technical report, National Aeronautics and Space Administration.

[4] Porretta, M., Dupuy, M.-D., Schuster, W., Majumdar, A., & Ochieng, W. (2008). Performance evaluation of a novel 4d trajectory prediction model for civil aircraft. *The Journal of Navigation*, *61*(3), 393–420.

[5] Yepes, J. L., Hwang, I., & Rotea, M. (2007). New algorithms for aircraft intent inference and trajectory prediction. *Journal of guidance, control, and dynamics*, *30*(2), 370–382.

[6] Schuster, W., Ochieng, W., & Porretta, M. (2010). High-performance trajectory prediction for civil aircraft. *29th Digital Avionics Systems Conference*, 1–C.

[7] Konyak, M., Doucett, S., Safa-Bakhsh, R., Gallo, E., & Parks, P. (2009). Improving ground-based trajectory prediction through communication of aircraft intent. *AIAA Guidance, Navigation, and Control Conference*, 6080.

[8] Gallo, E., Lopez-Leones, J., Vilaplana, M. A., Navarro, F. A., & Nuic, A. (2007). Trajectory computation infrastructure based on bada aircraft performance model. *2007 IEEE/AIAA 26th Digital Avionics Systems Conference*, 1–C.

[9] McGovern, S. M., Cohen, S. B., Truong, M., & Fairley, G. (2007). Kinematics-based model for stochastic simulation of aircraft operating in the national airspace system. *2007 IEEE/AIAA 26th Digital Avionics Systems Conference*, 3–B.

[10] Le Fablec, Y., & Alliot, J.-M. (1999). Using neural networks to predict aircraft trajectories. *IC-AI*, 524–529.

[11] Cheng, T., Cui, D., & Cheng, P. (2003). Data mining for air traffic flow forecasting: A hybrid model of neural network and statistical analysis. *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*, *1*, 211–215.

[12] De Leege, A., van Paassen, M., & Mulder, M. (2013). A machine learning approach to trajectory prediction. *AIAA Guidance, Navigation, and Control (GNC) Conference*, 4782.

[13] Hamed, M. G., Gianazza, D., Serrurier, M., & Durand, N. (2013). Statistical prediction of aircraft trajectory: Regression methods vs point-mass model. *ATM 2013, 10th USA/Europe Air Traffic Management Research and Development Seminar*, pp–xxxx.

[14] Rohani, A. S., Puranik, T. G., & Kalyanam, K. M. (2023). Machine learning approach for aircraft performance model parameter estimation for trajectory prediction applications. *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*, 1–9.

[15] Clausing, D. J. (1997). *Aviator's guide to navigation*. McGraw-Hill.

[16] Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Special issue on learning from imbalanced data sets. *ACM SIGKDD explorations newsletter*, *6*(1), 1–6.

[17] Ustuner, M., Sanli, F., & Abdikan, S. (2016). Balanced vs imbalanced training data: Classifying rapideye data with support vector machines. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *41*, 379–384.

[18] Landing AI. (2024). Data-centric ai [Accessed: 2024-01-19].

[19] Zha, D., Bhat, Z. P., Lai, K.-H., Yang, F., Jiang, Z., Zhong, S., & Hu, X. (2023). Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*.

[20] Kjærsgaard, R. D., Grønberg, M. G., & Clemmensen, L. K. (2021). Sampling to improve predictions for underrepresented observations in imbalanced data. *arXiv preprint arXiv:2111.09065*.

[21] Mani, I., & Zhang, I. (2003). Knn approach to unbalanced data distributions: A case study involving information extraction. *Proceedings of workshop on learning from imbalanced datasets*, *126*(1), 1–7.

[22] Tomek, I. (1976). Two modifications of cnn.

[23] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of artificial intelligence research*, *16*, 321–357.

[24] He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, 1322–1328.

[25] Olive, X., Basora, L., Viry, B., & Alligier, R. (2020). Deep trajectory clustering with autoencoders. *ICRAT 2020, 9th International Conference for Research in Air Transportation*.

[26] Deng, C., Kim, K., Choi, H.-C., & Hwang, I. (2021). Trajectory pattern identification for arrivals in vectored airspace. *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, 1–8.

[27] Deng, C., Choi, H.-C., Park, H., & Hwang, I. (2022). Trajectory pattern identification and classification for real-time air traffic applications in area navigation terminal airspace. *Transportation Research Part C: Emerging Technologies*, *142*, 103765.

[28] Murça, M. C. R. (2021). Identification and prediction of urban airspace availability for emerging air mobility operations. *Transportation Research Part C: Emerging Technologies*, *131*, 103274.

[29] Conde Rocha Murca, M., DeLaura, R., Hansman, R. J., Jordan, R., Reynolds, T., & Balakrishnan, H. (2016). Trajectory clustering and classification for characterization of air traffic flows. *16th AIAA Aviation Technology, Integration, and Operations Conference*, 3760.

[30] Sun, J., Ellerbroek, J., & Hoekstra, J. (2017). Flight extraction and phase identification for large automatic dependent surveillance–broadcast datasets. *Journal of Aerospace Information Systems*, *14*(10), 566–572.

[31] Olive, X., & Morio, J. (2019). Trajectory clustering of air traffic flows around airports. *Aerospace Science and Technology*, *84*, 776–781.

[32] Venturelli Cavalheiro, G., Vascik, P. D., Travnik, M., Salgueiro, S., & Hansman, J. (2021). Characterizing effective navigation performance of legacy aircraft to enable reduced procedural separation in support of advanced air mobility. *AIAA AVIATION 2021 FORUM*, 2389.

[33] Barratt, S. T., Kochenderfer, M. J., & Boyd, S. P. (2018). Learning probabilistic trajectory models of aircraft in terminal airspace from

position data. *IEEE Transactions on Intelligent Transportation Systems*, *20*(9), 3536–3545.

[34] Jung, S., & Kochenderfer, M. J. (2023). Inferring traffic models in terminal airspace from flight tracks and procedures. *arXiv preprint arXiv:2303.09981*.

[35] Zhu, X., Hong, N., He, F., Lin, Y., Li, L., & Fu, X. (2023). Predicting aircraft trajectory uncertainties for terminal airspace design evaluation. *Journal of Air Transport Management*, *113*, 102473.

[36] Krauth, T., Lafage, A., Morio, J., Olive, X., & Waltert, M. (2023). Deep generative modelling of aircraft trajectories in terminal maneuvering areas. *Machine Learning with Applications*, *11*, 100446.

[37] Kong, Y., & Mahadevan, S. (2023). Identifying anomalous behavior in aircraft landing trajectory using a bayesian autoencoder. *Journal of Aerospace Information Systems*, 1–9.

[38] Eerland, W. J., Box, S., & Sóbester, A. (2016). Modeling the dispersion of aircraft trajectories using gaussian processes. *Journal of Guidance, Control, and Dynamics*, *39*(12), 2661–2672.

[39] Shi, Z., Xu, M., Pan, Q., Yan, B., & Zhang, H. (2018). Lstm-based flight trajectory prediction. *2018 International joint conference on neural networks (IJCNN)*, 1–8.

[40] Ma, L., & Tian, S. (2020). A hybrid cnn-lstm model for aircraft 4d trajectory prediction. *IEEE access*, *8*, 134668–134680.

[41] Kim, H., & Lee, K. (2021). Air traffic prediction as a video prediction problem using convolutional lstm and autoencoder. *Aerospace*, *8*(10), 301.

[42] Choi, H.-C., Deng, C., & Hwang, I. (2021). Hybrid machine learning and estimation-based flight trajectory prediction in terminal airspace. *IEEE Access*, *9*, 151186–151197.

[43] Zeng, W., Quan, Z., Zhao, Z., Xie, C., & Lu, X. (2020). A deep learning approach for aircraft trajectory prediction in terminal airspace. *IEEE Access*, *8*, 151250–151266.

[44] Shi, Z., Xu, M., & Pan, Q. (2020). 4-d flight trajectory prediction with constrained lstm network. *IEEE transactions on intelligent transportation systems*, *22*(11), 7242–7255.

[45] Pang, Y., & Liu, Y. (2020). Conditional generative adversarial networks (cgan) for aircraft trajectory prediction considering weather effects. *AIAA Scitech 2020 Forum*, 1853.

[46] Jia, P., Chen, H., Zhang, L., & Han, D. (2022). Attention-lstm based prediction model for aircraft 4-d trajectory. *Scientific reports*, *12*(1), 15533.

[47] Yoon, S., & Lee, K. (2023). Improving aircraft trajectory prediction accuracy with over-sampling technique. *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*, 1–6.

[48] Choi, H.-C., Park, H., Deng, C., & Hwang, I. (2023). Multi-agent aircraft estimated time of arrival prediction in terminal airspace. *2023 IEEE/AIAA 42nd Digital Avionics Systems Conference (DASC)*, 1–9.

[49] Hong, S., & Lee, K. (2015). Trajectory prediction for vectored area navigation arrivals. *Journal of Aerospace Information Systems*, *12*(7), 490–502.

[50] Flightradar24. (2024). Real-time flight tracker — flightradar24 [Accessed: 2024-01-02]. https://www.flightradar24.com/37.46,126.43/13

[51] Fritsch, F. N., & Carlson, R. E. (1980). Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, *17*(2), 238–246.

[52] Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4). Springer.

[53] Gupta, M. R., Chen, Y. et al. (2011). Theory and use of the em algorithm. *Foundations and Trends® in Signal Processing*, *4*(3), 223–296.

[54] Kuha, J. (2004). Aic and bic: Comparisons of assumptions and performance. *Sociological methods & research*, *33*(2), 188–229.

[55] Schwarz, G. (1978). Estimating the dimension of a model. *The annals of statistics*, 461–464.

[56] Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, *37*(2), 233–243.

[57] Kramer, M. A. (1992). Autoassociative neural networks. *Computers & chemical engineering*, *16*(4), 313–328.

[58] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

[59] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735–1780.

[60] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, *27*.

[61] Müller, M. (2007). Dynamic time warping. *Information retrieval for music and motion*, 69–84.

[62] Rumelhart, D. E., Hinton, G. E., Williams, R. J. et al. (1985). Learning internal representations by error propagation.

[63] Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

[64] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, *32*.

[65] Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G. et al. (2017). Mixed precision training. *arXiv preprint arXiv:1710.03740*.

[66] Murça, M. C. R., & de Oliveira, M. (2020). A data-driven probabilistic trajectory model for predicting and simulating terminal airspace operations. *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, 1–7.

[67] Allendoerfer, K. R., Pai, S., & Friedman-Berg, F. J. (2008). The complexity of signal detection in air traffic control alert situations. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *52*(1), 54–58.

[68] Gong, C., & McNally, D. (2004). A methodology for automated trajectory prediction analysis. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 4788.

[69] Paglione, M., & Oaks, R. (2007). Implementation and metrics for a trajectory prediction validation methodology. *AIAA Guidance, Navigation and Control Conference and Exhibit*, 6517.

[70] Schuster, W., Porretta, M., & Ochieng, W. (2012). High-accuracy four-dimensional trajectory prediction for civil aircraft. *The Aeronautical Journal*, *116*(1175), 45–66.

[71] Ayhan, S., & Samet, H. (2016). Aircraft trajectory prediction made easy with predictive analytics. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 21–30.

Abstract in Korean

터미널 공역 내 항공기 궤적 복원 및 예측

윤석빈

스마트항공모빌리티학과

한국항공대학교 대학원

(지도교수: 이금진, Ph.D.)

궤적 예측은 항공교통관리의 중요한 요소 중 하나이다. 정확한 궤적 예측은 항공교통관리의 효율성을 높일 뿐만 아니라, 항공안전에도 큰 기여를 할 수 있다. 최근 정확한 궤적 예측 모델을 위해 인공지능과 다양한 데이터 기반 기법을 이용하여 연구가 이루어지고 있다. 하지만 이러한 기법들은 예측 모델의 훈련에 사용되는 데이터가 큰 영향을 미치는데, 데이터가 불충분하거나 불균형을 이루고 있다면 과적합 및 예측 정확도의 감소로 이어지게 된다. 본 논문에서는 전체 데이터셋 내에서 적은 비율을 차지하는 궤적 패턴이 궤적 예측 모델 개발에 있어 덜 중요하게 간주되어서는 안된다는 주장으로 새로운 접근법을 제안한다. 본 논문에서 제안하는 접근법은 Gaussian mixture model을 사용하여 궤적 패턴 식별을 수행하며, Transformer 구조에 기반한 궤적 복원 모델을 통해 전체 데이터셋 내 적은 비율을 차지하는 궤적 패턴에 대해 궤적 복원을 수행한다. 그 후 궤적 복원을 통해 증강된 데이터셋을 활용해 LSTM을 활용하는 Seq2Seq 네트워크를 훈련한다. 본 접근법은 인천국제공항에 착륙하는 실제 항공기의 항적 데이터를 사용하여 검증되었으며, 우수한 성능을 보임을 확인했다.

키워드: Air Traffic Management, Decision Support Tool, Data Augmentation, Trajectory Reconstruction, Trajectory Prediction, Gaussian Mixture Model, Transformer, Long short-term memory, Sequence to Sequence