Zurich University of Applied Sciences

School of Engineering

VT1

Stochastic Aircraft Trajectory Prediction

Author:
Alex Fustagueras

Supervisor:
Michael Felux
Manuel Waltert

A thesis submitted in fulfilment of the requirements for the degree of
Master of Science in Engineering MSE

at the

Centre for Aviation (ZAV)

October 28, 2025

Imprint

Project:	VT1
Title:	Stochastic Aircraft Trajectory Prediction
Author:	Alex Fustagueras
Date:	October 28, 2025
Keywords:	ADS-B, Aircraft Trajectory Prediction, Deep Learning, Long Short Term Memory, Mixture Density Network

Supervisor:
Michael Felux
Centre for Aviation (ZAV)
Zurich University of Applied Sciences

Study program:
Master of Science in Engineering MSE
Centre for Aviation (ZAV)
Zurich University of Applied Sciences

Abstract

The steady increase in European air traffic and the adoption of Free Route Airspace (FRA) highlight the need for reliable short-term aircraft trajectory prediction methods that also quantify inherent uncertainty. This study aims to evaluate the application of a Bidirectional Long Short-Term Memory (BLSTM) encoder–decoder architecture with a Mixture Density Network (MDN) output layer for probabilistic prediction of en-route aircraft trajectories. Historical Automatic Dependent Surveillance–Broadcast (ADS-B) data from Swiss airspace were preprocessed, filtered for en-route segments above flight level 19,500 ft, and transformed into structured training, validation, and test datasets. The model was trained to forecast short-term future positions based on recent trajectory history, using probabilistic Gaussian mixture outputs to represent uncertainty. Results show that while prediction errors increase with horizon length, the ground truth trajectories consistently remain within high-probability regions of the predicted distributions, confirming meaningful uncertainty quantification. A notable behaviour observed was the specialization of mixture components in modelling anisotropic uncertainty rather than distinct multimodal futures, reflecting the stable characteristics of en-route flight segments. Although the model's deterministic accuracy lags behind state-of-the-art benchmarks, the work demonstrates a viable proof-of-concept for stochastic trajectory prediction. The findings emphasize the importance of integrating environmental and contextual data, refining predictive accuracy, and exploring other structured probabilistic approaches as future research steps.

Contents

1. Introduction

1.1 Context

The European aviation sector is projected to grow significantly in air traffic volume, with EUROCONTROL forecasting a 3.7 % increase in the number of flights for 2025, reaching approximately 11 million flights movements and restoring pre-pandemic (2019) traffic levels by the second quarter of 2025 (EUROCONTROL 2025a). This upward trend is expected to continue, surpassing 12 million flights by 2031 and exhibiting an average annual growth rate of 2.2 % among states within the Single European Sky (SES) Performance Scheme. This expansion poses considerable challenges for Europe's en-route airspace, particularly in ensuring safety while accommodating the increased volume of traffic given the existing capacity limitations that restrict the number of flights that can be safely managed simultaneously.

To address these growing demands and ensure the safe and efficient flow of air traffic, the European Union has mandated, among other measures, the widespread implementation of Free Route Airspace (FRA) across all EU member states (European Parliament and Council 2011). FRA represents a paradigm shift in air traffic management, allowing airspace users to freely plan and fly their preferred trajectories directly between defined entry and exit points, without strict adherence to the traditional Air Traffic Service (ATS) route network (EUROCONTROL 2025b). This move towards user-preferred routing offers substantial benefits, including enhanced flight efficiency, reduced fuel consumption, decreased emissions, and potential cost savings for airlines (Tominaga et al. 2023). For example, route efficiency improvements can yield flight distance reductions ranging from 2 % to 3.5 %, potentially leading to notable financial savings for airlines across Europe (Brown et al. 2024).

Nevertheless, the increased flexibility introduced by FRA also brings inherent complexities, particularly in conflict detection and resolution (Çeçen and Çetek 2019). Unlike the predictable conflict "hotspots" of fixed ATS route airspaces, FRA disperses potential critical points throughout the entire airspace (SKYbrary Aviation Safety 2025). This means that loss of separation incidents can arise in virtually any location (Nava Gaxiola et al. 2018), making their anticipation and identification considerably more challenging for Air Traffic Controllers (ATCOs). While studies suggest that FRA might lead to a decrease in ATCO workload, for instance, by reducing the number of potential separation losses that require intervention (Tamara Pejovic 2019), the cognitive burden associated with continuously monitoring a more dynamic and less predictable traffic environment may, paradoxically, increase. Thus, the need for a proactive and continuous assessment of potential encounters across a less structured airspace requires data-driven approaches capable of accurately predicting future aircraft positions and their associated uncertainties in the en-route context. These predicted positions and uncertainties are crucial inputs for Conflict Risk Management (CRM) models, enabling the anticipation and mitigation of potential loss of separation, vital for maintaining safety in the evolving FRA environment.

1.2 State Of Knowledge

Given the ongoing relevance of aircraft trajectory prediction to ensure safety and efficiency in an increasingly complex airspace, a considerable amount of research has been conducted over the years to develop more sophisticated trajectory prediction models. Readers seeking a detailed review of aircraft trajectory prediction are referred to the comprehensive literature by Zeng et al. (2022). Nonetheless, this section offers a focused overview of the key concepts and approaches in the aircraft trajectory prediction field.

A trajectory is defined as an ordered sequence of time stamped spatial coordinates within a given reference frame. Trajectory prediction involves forecasting the four dimensional state vector of an aircraft encompassing latitude, longitude, altitude, and time based on an initial segment of its flight path (International Civil Aviation Organization 2005). The prediction can be categorized based on the forecast time horizon, which refers to the duration into the future for which a prediction is made. According to this timeframe, the purpose and characteristics of the prediction may differ. These can be classified as (1) short term predictions, focusing on the immediate future, essential for real time air traffic control, enabling tactical decision making to maintain safe separation between aircraft and resolve potential conflicts (Le et al. 2020); and (2) medium and long term forecasts, which support strategic air traffic management, airspace design, and traffic flow management tools that aim at forecasting the traffic demand for airspace resources to keep the capacity demand imbalance under limits. These longer term predictions aid in planning efficient routes and optimizing airspace capacity (Garcia-Chico et al. 2008).

To achieve these predictions, the literature mentions multiple approaches, which can be broadly divided into (i) kinetic models, (ii) state-estimation models, and (iii) data-driven models leveraging machine learning techniques. The applicability of these approaches can vary depending on the prediction's time horizon (i.e., short-term, medium-term, or long-term). Kinetic models rely on equations of motion and aerodynamic principles, requiring detailed aircraft performance parameters and environmental conditions. While highly accurate under ideal conditions, these models become sensitive to uncertainties in aircraft parameters and external factors such as wind and weather (Zeng et al. 2022). State-estimation models mathematically describe the system using state-space theory, propagating the aircraft's current state (i.e., position, speed, acceleration) into the future based on equations of motion and a state transition matrix (Ayala et al. 2023). Kalman Filters (KF) and Extended Kalman Filters (EKF) are prominent examples, along with Hidden Markov Models (HMMs), which have been used to correlate historical trajectories with environmental conditions, including weather, to predict future paths (Ayhan and Samet 2016; Yang et al. 2023). These models are simpler but may struggle with long-term accuracy due to uncaptured manoeuvring uncertainty.

The increasing availability of aircraft trajectory data, primarily from Automatic Dependent Surveillance-Broadcast (ADS-B) and Mode S, has facilitated the emergence of data-driven trajectory prediction models. These models leverage historical flight datasets to learn and predict future aircraft trajectories. Research has shown promising results using Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) and Bidirectional LSTM (BLSTM) architectures, to capture temporal dependencies within trajectory data (Yang et al. 2023). For instance, Pang et al. (2019) employed convolutional layers embedded within an LSTM architecture to integrate convective weather information, successfully improving trajectory predictions in weather-affected conditions. However, a significant portion of the literature primarily focuses on predicting a single, deterministic trajectory, often representing the most likely path (Ayala et al. 2023; Zeng et al. 2022). This approach, while useful, often fails to adequately capture the inherent uncertainty and variability in aircraft movements. Factors such as unexpected ATC instructions, minor deviations in flight plans, or the stochastic nature of wind and weather can lead to a distribution of possible future trajectories, rather than a single outcome. New approaches have explored probabilistic methods such as Gaussian Mixture Models (GMMs) within the data-driven framework; for example, Liu and Hansen (2018) utilized a Mixture Density Network (MDN) output layer to model a distribution of potential trajectory points conditioned on weather, and Rocha Murça and de Oliveira (2020) have also employed GMMs.

Despite the progress, the comprehensive and robust modelling of trajectory uncertainty of real-world flight operations remains an ongoing challenge, especially given dynamic weather or unexpected pilot manoeuvrers (Garcia-Chico et al. 2008). The limitations of deterministic predictions are critical for CRM systems in FRA, which require not only accurate position forecasts but also robust quantification of associated uncertainties to effectively assess and mitigate potential conflicts. Consequently, there remains a clear need for trajectory prediction methods capable of capturing complex, non-linear temporal dependencies in trajectory data while providing reliable estimates of prediction uncertainty, particularly in the challenging environment of en-route FRA operations in the short time range.

To address this limitations, Bidirectional-LSTM architectures with probabilistic decoders, such as Mixture Density Networks (MDNs), have emerged as powerful tools in other sequence prediction domains, including maritime trajectory prediction (Sørensen et al. 2022) and basketball trajectory prediction (Zhao et al. 2018). Their bidirectional nature allows them to learn from both past and future contextual information during training, enabling the capture of richer temporal dynamics. Furthermore, MDNs excel at modelling complex, multi-modal probability distributions, which allows for a more comprehensive representation of prediction uncertainty, particularly important in short-term forecasting. The effectiveness of this is illustrated by Zhao et al. (2018), who successfully predicted the ball's intricate trajectory dynamics throughout a basketball game. Therefore, the application of a BLSTM-MDN model to the prediction of aircraft trajectories offers a promising strategy to bridge this gap.

1.3 Research Objective

For this reason, this study aims to investigate how a BLSTM-MDN model can be applied to predict short-term aircraft trajectories by quantifying uncertainty in en-route airspace using historical ADS-B data. More specifically, the objectives of this research encompasses the development of a BLSTM-MDN model for short-term aircraft trajectory prediction, the evaluation of the model's performance in quantifying both prediction accuracy and its associated uncertainty within the en-route context, and the assessment of the contribution of using multiple mixture Gaussian distributions to probabilistic forecast quantification.

1.4 Thesis Structure

To guide the reader, the remainder of this paper is structured as follows: Section 2 outlines the methodology of this study, encompassing the model architecture, dataset, design, and training. Section 3 presents the results of the experiments, depicting the model's performance in trajectory prediction and uncertainty quantification, while Section 4 discusses the implications of the findings, comparing them with existing research and addressing the limitations of the study. Finally, Section 5 summarizes the key contributions and outlines potential directions for future work.

2. Methodology

This chapter outlines the methodological framework developed to create a model capable of predicting aircraft trajectories in an en-route context. The chapter is structured into two main sections: Section 2.1 details the data preparation steps undertaken to ensure data quality, consistency, and suitability for trajectory prediction tasks. Section 2.2 describes the neural network architecture and the training process using the curated dataset prepared in the previous section.

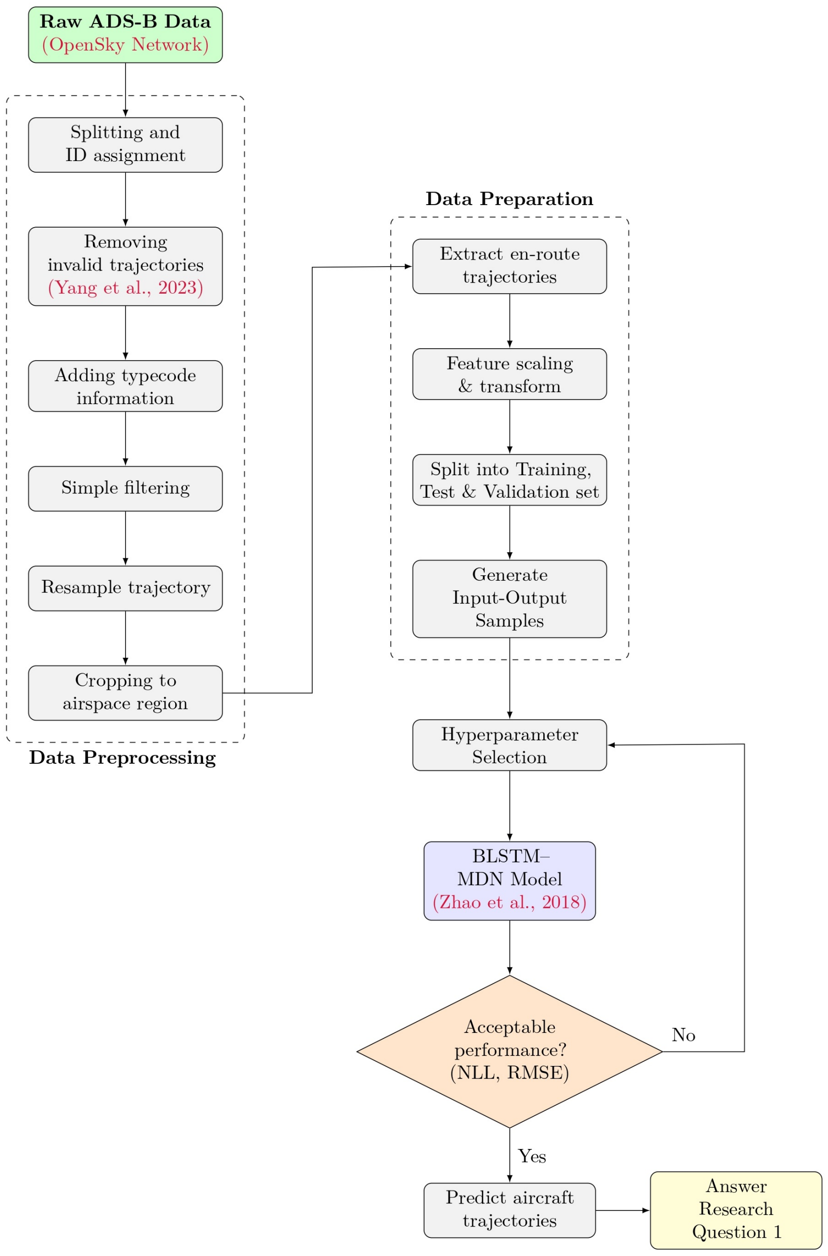

Raw ADS-B Data
(OpenSky Network)
Splitting and ID assignment
Removing invalid trajectories (Yang et al., 2023)
Adding typecode information
Simple filtering
Resample trajectory
Cropping to airspace region
Data Preprocessing
Data Preparation
Extract en-route trajectories
Feature scaling & transform
Split into Training, Test & Validation set
Generate Input-Output Samples
Hyperparameter Selection
BLSTM–MDN Model (Zhao et al., 2018)
Acceptable performance? (NLL, RMSE)
No
Yes
Predict aircraft trajectories
Answer Research Question 1

FIGURE 2.1: Schematic overview of the research methodology adopted throughout the study.

2.1 Data Processing

Since the methods proposed in this study strongly rely on the accurate and comprehensive use of ADS-B trajectory data, ensuring data quality is vital for obtaining reliable and robust prediction results. Therefore, prior to detailing the design of the model, a well-prepared dataset must be constructed. This process involves several stages: Section 2.1.1 outlines the initial acquisition of raw ADS-B trajectory data; Section 2.1.2 details the preprocessing steps applied to the acquired data; Section 2.1.3 covers the identification of relevant en-route segments; Section 2.1.4 elaborates on the feature engineering techniques employed to enhance the dataset's informational content and utility; and finally, Section 2.1.5 presents the generation of the structured dataset tailored for model training and evaluation.

2.1.1 Data Fetching

The raw trajectory data for this methodology are derived from ADS-B transmissions, containing essential flight information for trajectory forecasting. These data were obtained from the OpenSky Network historical database (Schäfer et al. 2014), an extensive open-access repository of flight trajectory information. Data collection focused on flights intersecting the Flight Information Region (FIR) of the Swiss Airspace (FRA-LSAS), whose boundaries are shown in Figure 2.2. The collection period extended from July 1 to July 31, 2024, with daily acquisition in raw Apache Parquet format, which facilitates efficient subsequent processing.

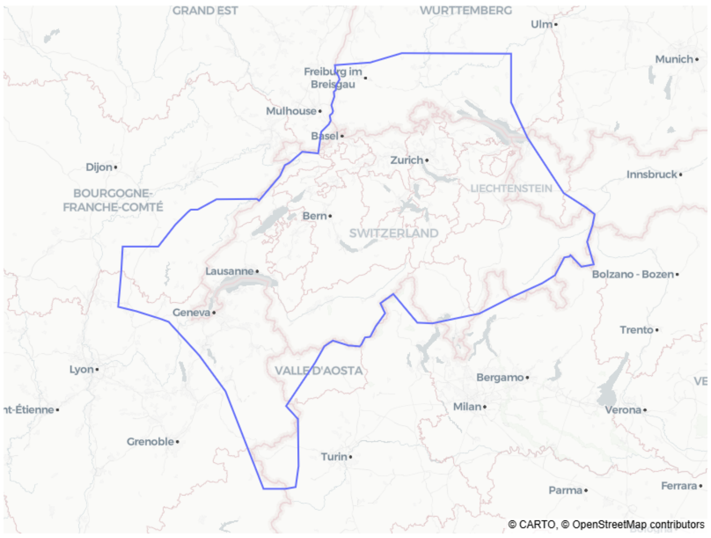


FIGURE 2.2: Geographical boundaries of the defined FRA-LSAS airspace.

2.1.2 Data Preprocessing

Following raw data acquisition, several preprocessing procedures were implemented using the `traffic` Python library by Olive (2019). Each daily package underwent a series of cleaning and enrichment operations to prepare it for analysis. These operations included the assignment of unique flight identifiers, using built-in functions such as `assign_id()`, and the removal of anomalously short and ground-only flights. Additionally, median and mean filtering were applied to eliminate outliers and improve signal quality for key trajectory parameters such as altitude, vertical rate, groundspeed, and track. To further refine the data, geographical clipping strictly constrained trajectories within the defined FRA-LSAS airspace boundaries, while resampling of trajectory data ensured consistent temporal intervals of 1 second between consecutive data points. Once these cleaning procedures were completed, an aggregated file containing all relevant trajectories was obtained for further usage.

2.1.3 Identification and Extraction of En-route Trajectories

The aggregated raw data file initially contained a total of 379,409 flights. Since the focus of this study is on en-route trajectory prediction, it is crucial to establish clear criteria for isolating flight segments pertinent to this task. The selection of these criteria is driven by the need to distinguish the stable, high-altitude cruising phase from climb and descent operations.

To illustrate the application of this criteria, Figure 2.3 exemplifies how a specific altitude threshold can be applied to a single flight trajectory to identify its en-route portion. The shaded green area visually highlights the segment where the flight's altitude meets or exceeds the defined minimum for en-route operations.

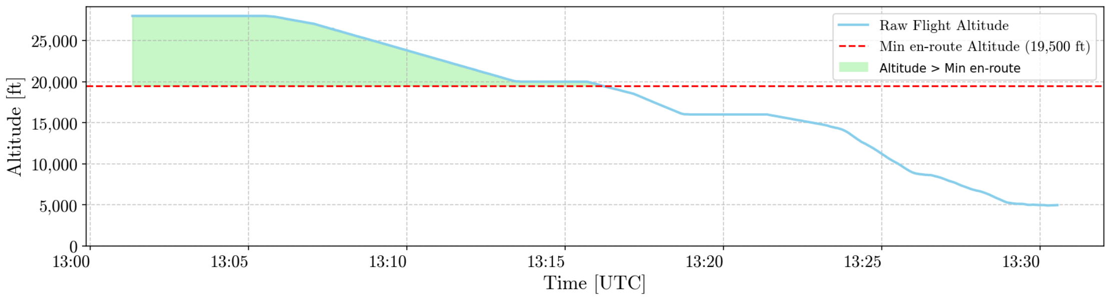


FIGURE 2.3: Example of a single flight altitude's profile with the 19,500 ft minimum en-route altitude threshold applied. The shaded green area indicates the identified en-route portion.

For this study, en-route trajectory segments were primarily identified based on an altitude criterion. A minimum altitude of 19,500 feet (FL195) was established as the lower bound for en-route flight. This specific threshold aligns with established aviation standards and practices in Switzerland, where 19,500 feet is explicitly defined as the altitude separating low and high airway structures, indicating a common transition level for en-route operations (FABEC 2022).

The selection of FL195 is further supported by the overall altitude distribution of the raw trajectory data. As shown in Figure 2.4, there is a clear distinction between lower altitude operations (climb/descent) and the higher-altitude cruising phase. The chosen threshold effectively separates these distinct flight regimes, ensuring that the extracted segments represent stable en-route conditions.

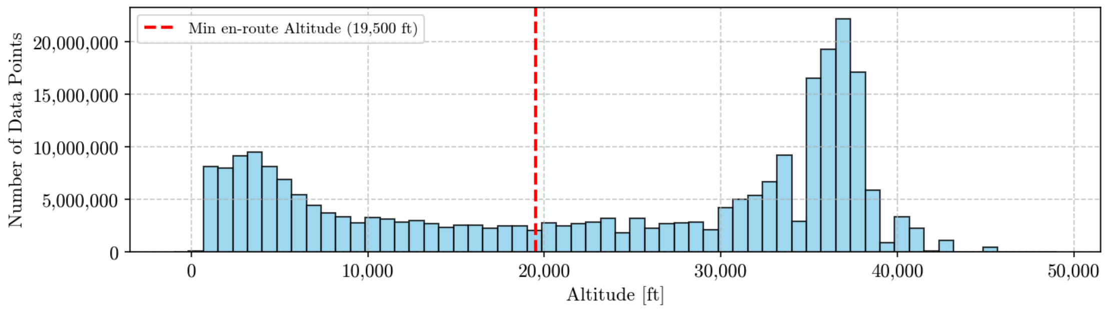


FIGURE 2.4: Distribution of raw trajectory data points by altitude, with the 19,500 ft minimum en-route altitude threshold indicated.

This extraction process was parallelized to optimize computational efficiency, yielding a refined set of trajectories comprising exclusively relevant en-route flight segments. Following these filtering steps, a total of 180,519 flights were retained.

2.1.4 Data Preparation

To prepare the preprocessed flight data for model training, feature engineering and scaling were performed. This process transforms raw observations into a suitable format for machine learning algorithms, enhancing model performance and learning stability.

Categorical Variable Encoding

The aircraft typecodes, inherently categorical, were transformed into a numerical representation using one-hot encoding. This method creates binary columns for each unique typecode, which is crucial for preventing the model from inferring false ordinal relationships from nominal data. To ensure focus on statistically significant aircraft types, only those with over 1,500 flights were selected for encoding. This threshold helps reduce feature space dimensionality and allows the model to learn from frequently observed flight behaviours. Following this restriction, 141,659 flights remained for subsequent processing.

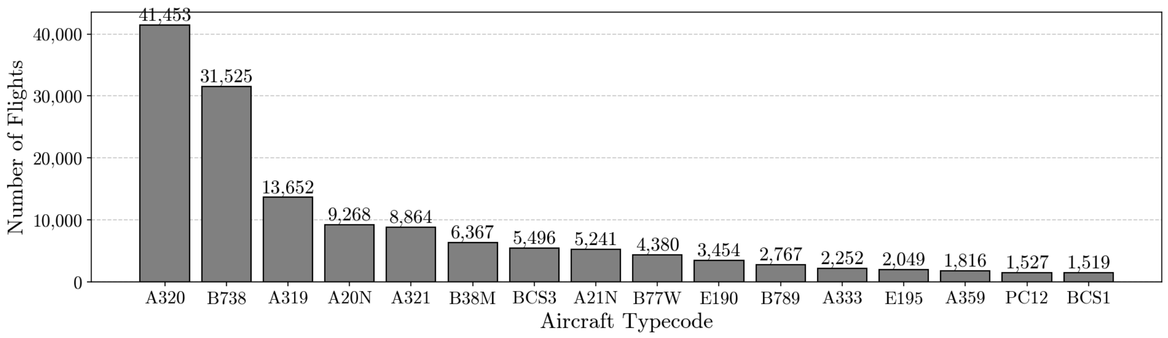


FIGURE 2.5: Distribution of the number of flights for each selected aircraft typecode, illustrating the frequency of different aircraft types in the dataset.

Numerical Feature Engineering

To enrich the dataset with dynamic characteristics trying to make more accurate trajectory prediction, several new numerical features were derived from the raw trajectory data:

- **Heading rate:** This quantifies the rate of change in the aircraft's track angle over time. Can result useful for anticipating turns and manoeuvrers during the en-route phase.
- **Groundspeed change:** This metric represents the acceleration or deceleration along the aircraft's ground track, for enhancing the model at speed adjustments.
- **Vertical acceleration:** Derived from changes in vertical rate, this feature captures the rate at which an aircraft changes its vertical speed; might be powerful for anticipating subtle altitude adjustments, even in stable en-route flight.

Feature Scaling

A Min-Max scaler was applied to normalize all numerical features to a standardized range between 0 and 1. This prevents features with larger numerical ranges from disproportionately influencing the model and promotes faster, more stable learning. The scaling was performed using the following formula:

$$x_{\text{scaled}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

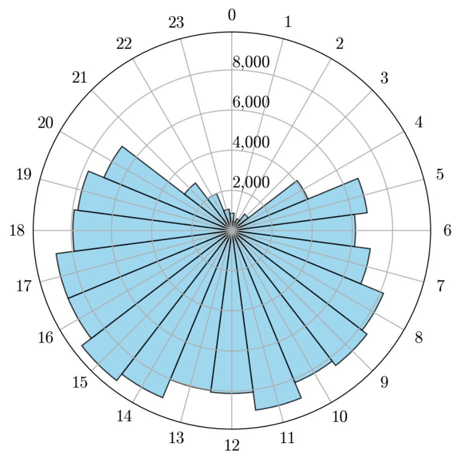
Separate scalers were applied for input features (e.g., latitude, longitude, altitude, vertical rate, groundspeed, heading rate, groundspeed change, vertical acceleration) and output features (latitude, longitude, altitude). This distinction is critical for maintaining prediction-specific scaling, ensuring that the model's outputs can be accurately inverse-transformed to their original meaningful scale.

Temporal Feature Encoding

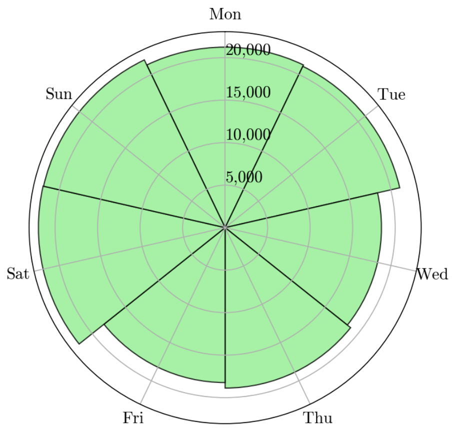
Cyclical variables, such as the time of day (hour) and day of the week (weekday), required a specialized encoding to preserve their periodic nature and continuity. Standard normalization methods can create artificial discontinuities (e.g., 23:00 and 00:00 appearing numerically distant). Therefore, these temporal features were encoded using sine and cosine transformations, mapping a single cyclical variable into two continuous features. For this study, month data was not included as the dataset pertained to a single month of operations. The formula used to encode a cyclic variable (x) within a given period (P) is:

$$x_{\sin} = \sin\left(2\pi\frac{x}{P}\right) \qquad x_{\cos} = \cos\left(2\pi\frac{x}{P}\right)$$

This transformation ensures that numerically close times (e.g., 23:00 and 00:00) also have close representations in the transformed feature space, facilitating the model's ability to learn consistent temporal patterns.



(A) Distribution of flights for hours of the day.



(B) Distribution of flights for days of the week.

FIGURE 2.6: Distribution of the number of flights on the temporal features after applying sine and cosine transformations.

2.1.5 Generation of Training, Validation, and Test Datasets

Once all features have been enhanced, the next crucial step involved preparing data samples for the prediction model. Prior to generating these samples, it was necessary to partition the dataset into distinct training, validation, and test sets. This division was performed at the trajectory level, identified by unique `flight_id`, ensuring that each set comprises complete flight trajectories, preventing any single flight from being inadvertently split across subsets. Such a separation is paramount to avoid evaluation on trajectory segments already encountered during training, which would compromise the unbiased assessment of its performance on unseen data. The dataset was then randomly partitioned using a ratio of 60% for training, 20% for validation, and 20% for testing, yielding 81,996 training, 29,587 validation, and 30,076 test flights.

Input-Output Sample Generation

Subsequently, structured input-output samples were generated from each trajectory within the partitioned sets using an iterative sampling process. Each sample comprises an input sequence, representing a past observation window, and a corresponding output sequence, which the model is trained to predict, enabling the model to learn the temporal dynamics of flight trajectories. This sampling process, visually detailed in Figure 2.7, extracts up to a maximum number of samples m per flight by spacing the desired number of samples across the entire flight. Key parameters defining this generation are:

- **Input sequence duration: 24 seconds (sampled at 1-second intervals)** – This defines the historical trajectory segment provided to the model.
- **Output sequence duration: 60 seconds (sampled at 1-second intervals)** – This specifies the future trajectory segment the model is trained to forecast. The output sequence duration of 60 seconds was specifically chosen to align with typical requirements for short-term trajectory forecasting in Air Traffic Management (ATM) applications, focusing on the immediate future trajectory while balancing computational feasibility.
- **Overlap between input and output sequences: 1 second** – An overlap was introduced, where the final time step of the input sequence serves as the initial time step of the corresponding predicted output sequence. This approach was adopted to improve the continuity between the input and the corresponding prediction.

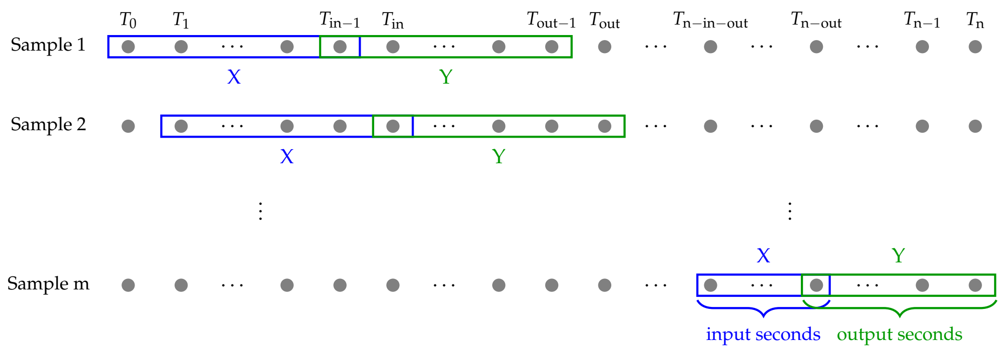


FIGURE 2.7: Schematic of the iterative process used to generate input-output sample pairs along the en-route trajectories.

The sample generation procedure was applied separately to all flights within the sets. The resulting arrays were then concatenated to form a single input and output array per set.

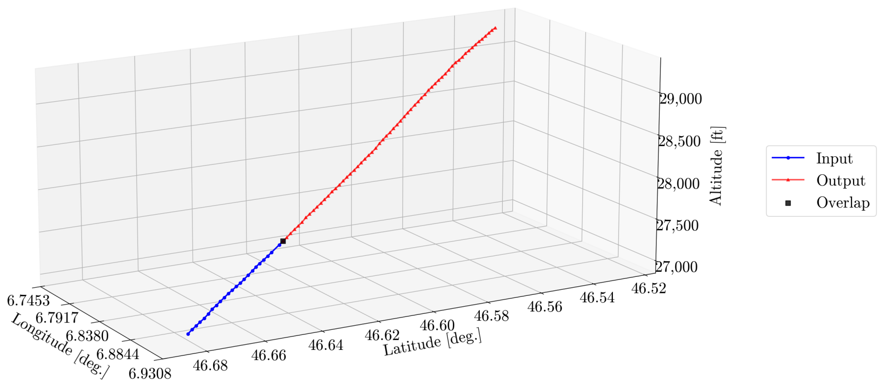


FIGURE 2.8: Visualization of a generated input-output sample pair, depicting the input sequence (observed) and the output sequence (to be predicted).

Input Feature Organisation

To optimise model architecture and efficiency, it was decided to split the input features into two distinct arrays: time-variant (variable) and time-invariant (constant) parameters. While all input features were initially provided over the input duration, many parameters such as aircraft typecode or day of the week are constant. Providing these features repeatedly across all time steps is unnecessary and inefficient. Accordingly, the input arrays for each set (training, validation, test) were structured as follows:

- **Variable input features** (12 dimensions, across the 24 time steps): These included scaled latitude, longitude, altitude, vertical rate, groundspeed, heading rate, groundspeed change, vertical acceleration, track sine/cosine, and hour sine/cosine. These features inherently vary over time.
- **Constant input features** (18 dimensions, representing a single point in time): These comprised the one-hot encoded aircraft typecodes and the cyclical weekday sine/cosine. For these features, only the data corresponding to the first time step of the input sequence was retained, as their values remain constant within the input window.

This organisation resulted in final sample shapes that differentiate between time-varying trajectory data and static flight characteristics. The number of resulting input-output sample pairs are 3,922,159 for training, 1,328,057 for validation, and 1,385,700 for testing.

2.2 Model Design and Training

Upon having processed the data, the next step is training the predictive model on the resulting dataset. Details of this procedure are discussed below, with Section 2.2.1 outlining the selection of the model architecture, Section 2.2.2 covering the definition of further hyperparameters, and Section 2.2.3 providing insights into the actual training process.

2.2.1 Model Architecture

To effectively capture the complex dependencies inherent in trajectory prediction and to provide a probabilistic forecast, an encoder-decoder architecture was chosen, incorporating a Mixture Density Network (MDN) at its output. This design is particularly well-suited for sequence-to-sequence problems and for modelling multi-modal distributions, offering a richer understanding of future possibilities compared to deterministic predictions.

The model operates on a dual-input and single-output framework, addressing both time-variant and time-invariant features. The time-invariant features are first expanded to match the sequence length of the time-variant inputs. This ensures that the constant contextual information is available at every time step, providing consistent context for the subsequent recurrent layers. The expanded time-invariant features are then concatenated with the time-variant features, forming a unified input sequence for the encoder. A Bidirectional Long Short-Term Memory (BLSTM) network serves as the encoder. The choice of a bidirectional recurrent network is crucial for capturing dependencies from both forward and backward directions within the input historical window. This allows the model to form a comprehensive understanding of the past trajectory by processing information from both earlier and later time steps in the input sequence. The encoder consists of one BLSTM layer with a tunable number of units in each direction (forward and backward). In this study, values such as 64, 128, and 256 units per direction were explored, resulting in encoder state vectors of 128, 256, and 512 dimensions after concatenation. This configuration directly influenced the size of the decoder, which was typically set to match the dimensionality of the concatenated encoder states. The final hidden and cell states from both the forward and backward LSTMs are extracted and concatenated, producing a rich contextual summary of the input sequence.

The decoder is an LSTM network responsible for generating the future trajectory sequence. It is initialised with the concatenated hidden and cell states from the encoder, enabling the transfer of learned contextual information from the historical input to the future prediction. Similar to the encoder's input preparation, the time-invariant features and the encoder's hidden state are repeated to be available at each output time step. This conditional setup allows the decoder to generate the future trajectory sequence, which has a length of 61 time steps, based on the learned context. The decoder consists of a single LSTM layer with a number of units equal to the dimensionality of the concatenated encoder states, and is configured to return sequences at each time step.

To facilitate probabilistic trajectory prediction, the output of the decoder is fed into a Mixture Density Network (MDN). Unlike standard neural networks that output single point estimates, an MDN models the conditional probability distribution of the output, allowing for the representation of uncertainty and multi-modal outcomes. This is particularly advantageous in scenarios where multiple future trajectories are plausible. A `TimeDistributed` wrapper is applied to the MDN output layer, ensuring that the MDN is independently applied to the output of each time step predicted by the LSTM decoder. The MDN outputs parameters for a Gaussian Mixture Model (GMM) at each predicted time step. Specifically, for each of the 3 output features (latitude, longitude, altitude), and for each predicted mixture component (K components), the MDN predicts:

- **Means (μ):** The central locations of each Gaussian component.
- **Standard Deviations (σ):** The spread of each Gaussian component. An exponential activation function is applied to ensure that standard deviations are positive.
- **Mixing Coefficients (π):** The probabilities or weights associated with each Gaussian component, which sum to one across all components.

The probability density function (PDF) of a Gaussian Mixture Model for an output $\mathbf{y}$ conditioned on input $\mathbf{x}$ is given by:

$$p(\mathbf{y}|\mathbf{x}) = \sum_{k=1}^{K} \pi_k(\mathbf{x}) \mathcal{N}(\mathbf{y}|\boldsymbol{\mu}_k(\mathbf{x}), \boldsymbol{\Sigma}_k(\mathbf{x})) \tag{2.1}$$

where K is the number of mixture components, $\pi_k(\mathbf{x})$ are the mixing coefficients, $\boldsymbol{\mu}_k(\mathbf{x})$ are the means, and $\boldsymbol{\Sigma}_k(\mathbf{x})$ are the covariance matrices for each component k.

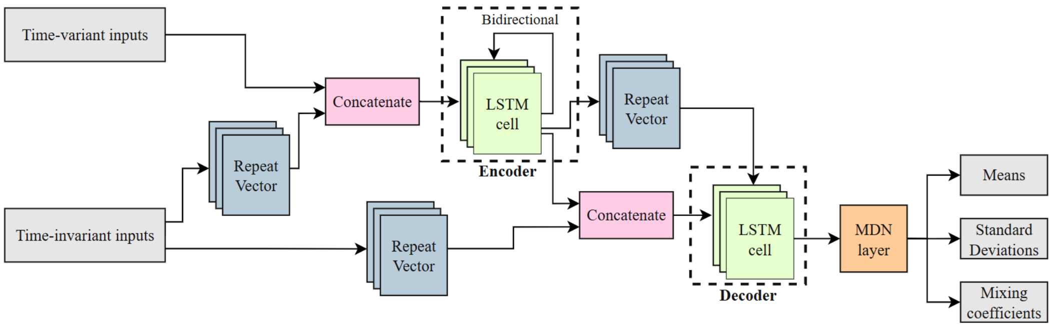


FIGURE 2.9: A more detailed representation of the chosen model architecture, including information on the types of layers used.

2.2.2 Other Hyperparameters

The model's behaviour is further defined by hyperparameters that control the learning process and network configuration. The default hyperbolic tangent (tanh) activation function was used for all LSTM layers. For the standard deviations output by the MDN, an exponential activation function was applied to ensure positivity. A custom loss function was defined to optimise the model, combining two objectives:

1. **Negative Log Likelihood (NLL) of the Mixture:** This is the primary component of the loss function, directly optimising the likelihood of the true trajectory given the predicted Gaussian Mixture Model. Minimising the NLL encourages the model to assign higher probability to the actual observed trajectories. Conceptually, this is defined as:

$$L_{\text{NLL}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\pi_k(\mathbf{x}^{(i)})\mathcal{N}(\mathbf{y}^{(i)}|\boldsymbol{\mu}_k(\mathbf{x}^{(i)}),\boldsymbol{\Sigma}_k(\mathbf{x}^{(i)}))\right)$$

   where N is the number of samples, $\mathbf{y}^{(i)}$ is the true output, $\mathbf{x}^{(i)}$ is the input, and K, π_k, $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ are as defined for the GMM PDF in Equation 2.1.

2. **Weighted Mean Squared Error (MSE) Term:** An additional weighted MSE term was included between the true trajectory and the weighted mean of the predicted Gaussian components. Its inclusion can provide a deterministic "anchor" for the probabilistic prediction, guiding the optimisation process. The weighted mean of the prediction $\hat{\mathbf{y}}_{\text{mean}}$ for a given input is:

$$\hat{\mathbf{y}}_{\text{mean}} = \sum_{k=1}^{K}\pi_k(\mathbf{x})\boldsymbol{\mu}_k(\mathbf{x}) \qquad \text{MSE term is then } L_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}||\hat{\mathbf{y}}^{(i)}_{\text{mean}} - \mathbf{y}^{(i)}||^2$$

The total loss function is a weighted sum of these two components: $L_{\text{Total}} = L_{\text{NLL}} + \lambda \cdot L_{\text{MSE}}$

For optimizing the model's parameters, the Adam optimizer was selected for its known efficiency in handling sparse gradients and its adaptive learning rate capabilities, making it particularly well-suited for training complex recurrent neural networks. An initial learning rate of 1×10^{-3} was often used, with variations and adaptive reductions explored as part of the tuning process. To further stabilise training, especially for recurrent neural networks, gradient clipping with a `clipnorm` value of 1.0 was applied. A batch size of 512 samples was determined to be effective after initial tests and consistently used across the primary experiments.

The iterative process for tuning these parameters involved systematic experimentation and rigorous performance analysis. This approach was vital for understanding the impact of each hyperparameter on the model's convergence, stability, and ultimate predictive performance. Specifically, different values were explored for the number of encoder units (e.g., 64, 128, and 256 units per direction), which directly influenced the corresponding decoder size (e.g., 128, 256, and 512 units for the decoder, respectively). Other tuned parameters included the number of mixture components K in the MDN (e.g., 1, 3), and the weighting factor λ in the total loss (ranging from 0.0 to 0.2, and potentially up to 0.5 in broader explorations). The sequence lengths of 24-second input history, 1-second overlap, and 60-second output prediction were fixed throughout all experiments, as they reflect the inherent structure of the dataset and not tunable hyperparameters. Detailed evaluations of these variations are provided in Section 3.

2.2.3 Model Training

Models were trained for a maximum of 50 epochs, their performance evaluated on the entire validation dataset at the end of each epoch. Data was efficiently handled by loading large datasets using memory-mapping directly into NumPy arrays, with TensorFlow's Keras API internally managing batching for training and validation from these loaded arrays.

To optimize the training process and prevent overfitting, several callbacks were implemented, with their configurations refined based on observed training dynamics:

- `ModelCheckpoint` saved the best model weights based on validation set performance, ensuring strong generalization.
- `EarlyStopping` halted training if validation loss did not improve for 10 consecutive epochs, balancing training efficiency with robust convergence.
- `ReduceLROnPlateau` adaptively reduced the learning rate by a factor of 0.5 if validation loss stagnated for 5 epochs, aiding in fine-tuning and navigating complex loss landscapes.

Real-time monitoring and visualisation of key metrics, such as the unscaled Mean Squared Error (MSE) for X, Y, and Z, were provided through a custom `LogValMSE` callback. This callback was explicitly integrated with the Weights & Biases toolkit for comprehensive experiment tracking, allowing for easy tracking of each training run's progress and the impact of different hyperparameter choices, as seen in Figure 2.10. This systematic tracking was indispensable for iterating on model improvements and identifying optimal configurations.

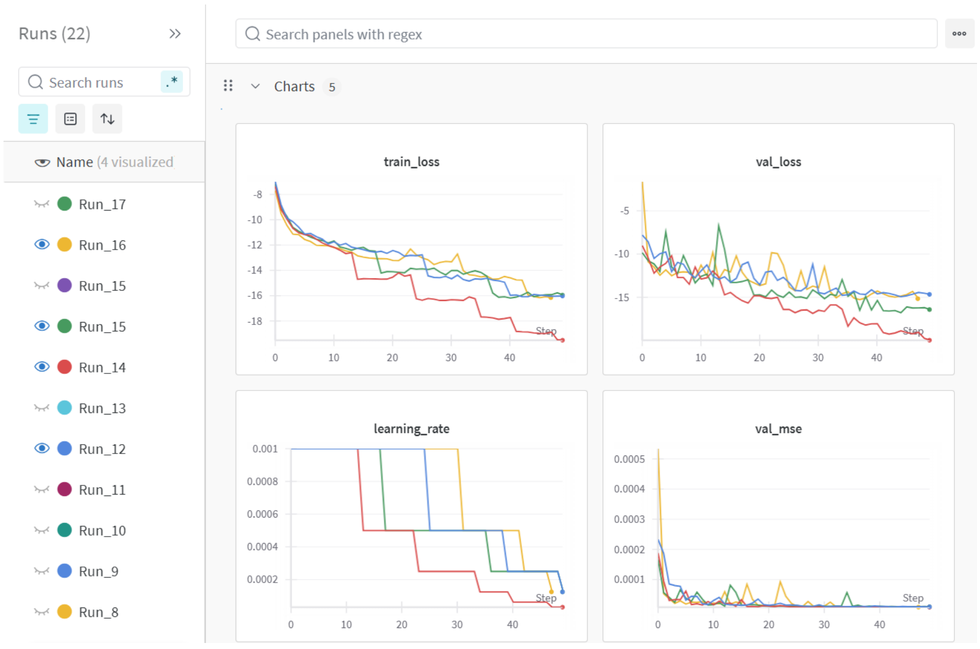


FIGURE 2.10: Screenshot of the Weights & Biases Dashboard used for tracking each training run progress.

Training was conducted on a computational environment leveraging both CPU resources and a Tesla V100-SXM2-32GB GPU, chosen for its computational power and memory capacity, which facilitated faster iteration and enabled the exploration of a wider range of hyperparameters and larger batch sizes. Upon completion of training, the model's weights, architecture, configuration, and comprehensive training history were saved in multiple formats for robustness and future compatibility.

3. Results

This section presents the empirical results obtained from the trained trajectory prediction model. Following a systematic hyperparameter tuning process involving 20 distinct training runs shown in Figure 2.10, one configuration (Run #14) was identified as the best-performing model. This selection was primarily based on the lowest Negative Log-Likelihood (NLL) validation, which reflects the model's ability to produce sharp and calibrated probabilistic forecasts. As a secondary criterion, the unscaled Mean Squared Error (MSE) was also monitored to ensure that deterministic prediction accuracy remained competitive.

Best Model Configuration (Run #14):

- **Mixture components:** $K = 3$ (Mixture Density Network output)
- **Loss function:** Weighted sum of NLL and unscaled MSE with $\lambda = 0.2$
- **Encoder / Decoder:** One-layer bidirectional LSTM with 64 hidden units per direction
- **Latent dimension:** 128 (after bidirectional concatenation)
- **Learning rate schedule:** `ReduceLROnPlateau` with patience of 5 epochs and factor of 0.5, reducing from an initial value of 1×10^{-3} to a final value of 3.125×10^{-5}

This configuration was trained for up to 50 epochs with early stopping based on validation NLL, and all training metrics were logged using the Weights & Biases platform. Run #14 showed not only the lowest validation NLL (see Figure 2.10) but also stable training curves with minimal overfitting. Details of the training metrics are visualized in Figure 3.1.

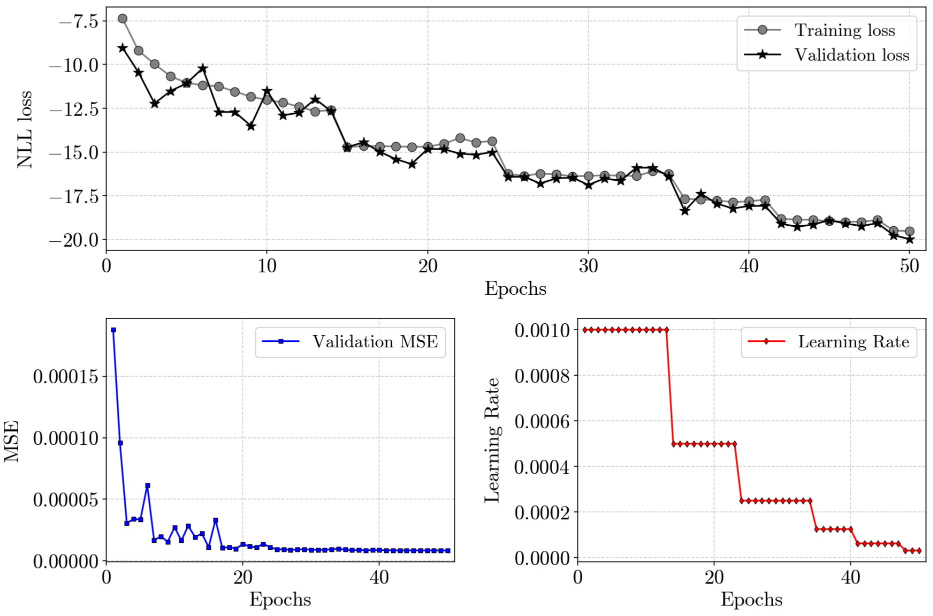


FIGURE 3.1: Training and validation metrics for Run #14. The plot shows the evolution of NLL and unscaled MSE across epochs. The model exhibits stable convergence, with the learning rate adapting via the scheduling.

The subsequent analysis focuses exclusively on the performance of Run #14, and includes both quantitative metrics and qualitative visualizations that demonstrate the model's probabilistic capabilities. All results are presented in the Earth-Centered, Earth-Fixed (ECEF) coordinate system, with positions and errors expressed in meters.

3.1 Quantitative Analysis of Prediction Errors

The model's predictive performance was first assessed by evaluating the absolute errors in ECEF coordinates (X, Y, Z) on the unseen test dataset. Figure 3.2 illustrates the distribution of these absolute errors across the 61-second prediction horizon.

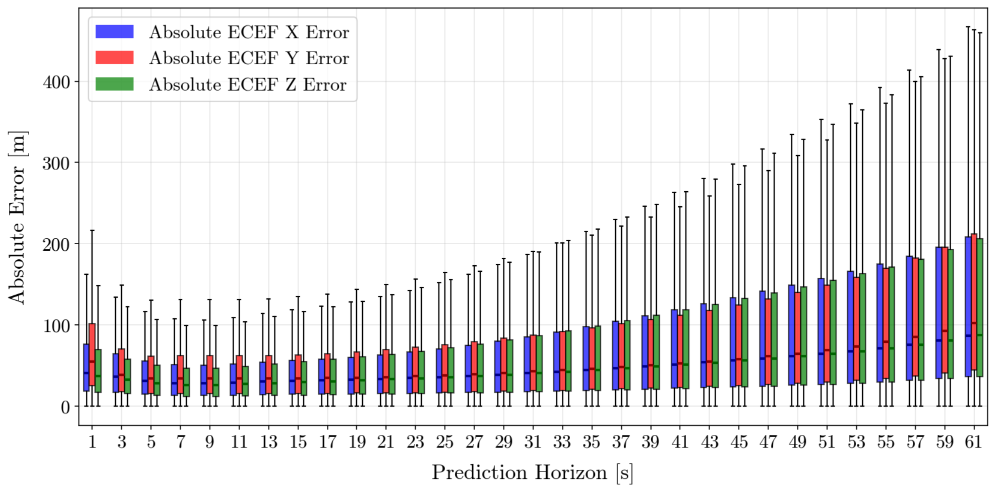


FIGURE 3.2: Box plots of absolute X, Y, and Z errors across the time horizon for all samples in the test dataset. The boxes represent the interquartile range (IQR), and the whiskers extend to 1.5× IQR.

A general trend of increasing error with the prediction horizon is observable across all coordinates at least from beyond 7-9 seconds. While median errors are modest for shorter horizons, they grow as the forecast extends. The X component consistently exhibits slightly higher median errors compared to the Y and Z components at longer horizons. A notable characteristic is the high error in the Y direction at the first time step, which subsequently decreases. Furthermore, the maximum error range, indicated by the whiskers, follows a parabolic trend, reaching a local minimum at approximately 8 seconds.

To provide a more granular view, the prediction horizon was segmented into early (1–20 s), mid (21–40 s), and late (41–61 s) stages, with mean absolute errors as follows:

1. **Early horizon (1–20 s):** 61.7 m (X), 82.9 m (Y), and 56.8 m (Z).

2. **Mid horizon (21–40 s):** 78.8 m (X), 87.0 m (Y), and 78.7 m (Z).

3. **Late horizon (41–61 s):** 199.5 m (X), 225.3 m (Y), and 193.8 m (Z).

Across the entire prediction horizon, the overall mean absolute errors were 94.5 m (X), 104.2 m (Y), and 92.1 m (Z). These figures confirm that while the Z error is initially the lowest, the Y error experiences the most significant degradation over time, becoming the largest source of error in the late horizon and overall.

3.2 Qualitative Analysis of Probabilistic Predictions

Beyond overall error statistics, Mixture Density Network's key strength lies in its ability to provide probabilistic forecasts, quantifying uncertainty. This can be demonstrated through visualizations of an individual sample predictions and the characteristics of its predicted distributions.

Figure 3.3 presents a detailed view of a single sample's trajectory prediction (Sample #1199302). This sample was selected as it demonstrates the model's ability to capture a gradual shift in the trajectory within its probabilistic forecast.

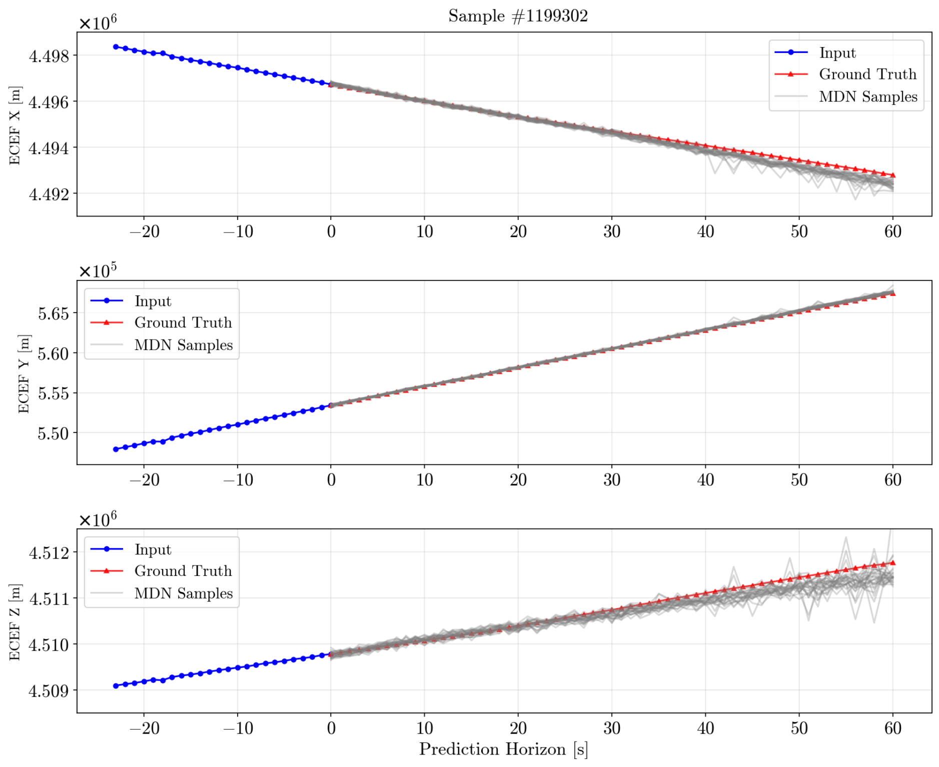


FIGURE 3.3: Trajectory visualization showing the input history (blue), ground truth trajectory (red), and multiple samples (grey) drawn from the predicted MDN distribution across the prediction horizon for X, Y, and Z coordinates.

As shown in Figure 3.3, the trajectories sampled from the MDN distribution cluster around the ground truth path. The dispersion among these samples widens as the prediction horizon increases, reflecting the model's growing predictive uncertainty over time.

To further quantify the error for this specific sample, Figure 3.4 plots the absolute error for each coordinate over the prediction horizon.

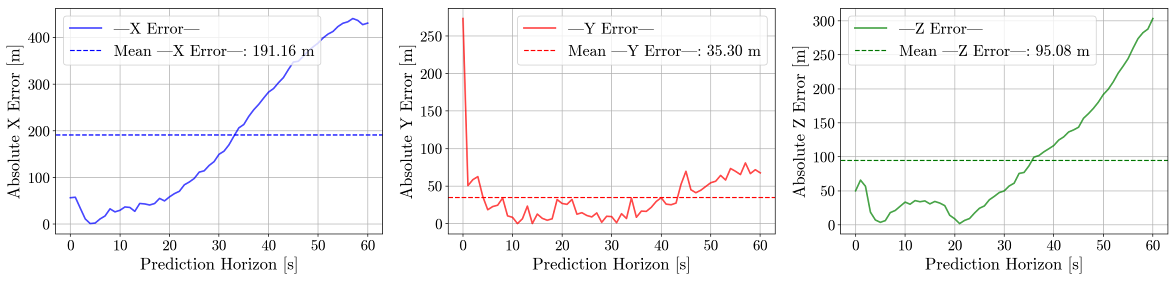


FIGURE 3.4: Absolute errors for X, Y, and Z coordinates over the prediction horizon for Sample #1199302. The dashed lines indicate the mean absolute error for each coordinate across the horizon for this specific sample.

Figure 3.4 confirms that the errors in X and Z exhibit a distinct increasing trend at further horizon times, while the Y error remains comparatively low and stable. The first timestamp again displays an elevated Y error, consistent with the pattern observed in the aggregate results from Figure 3.2. However, for this specific sample, the mean Y error is the lowest among all three axes, diverging from the overall dataset behaviour.

The probabilistic nature of the MDN output is further detailed in Figure 3.5, which displays the spatial distribution of predicted samples in the X-Y plane.

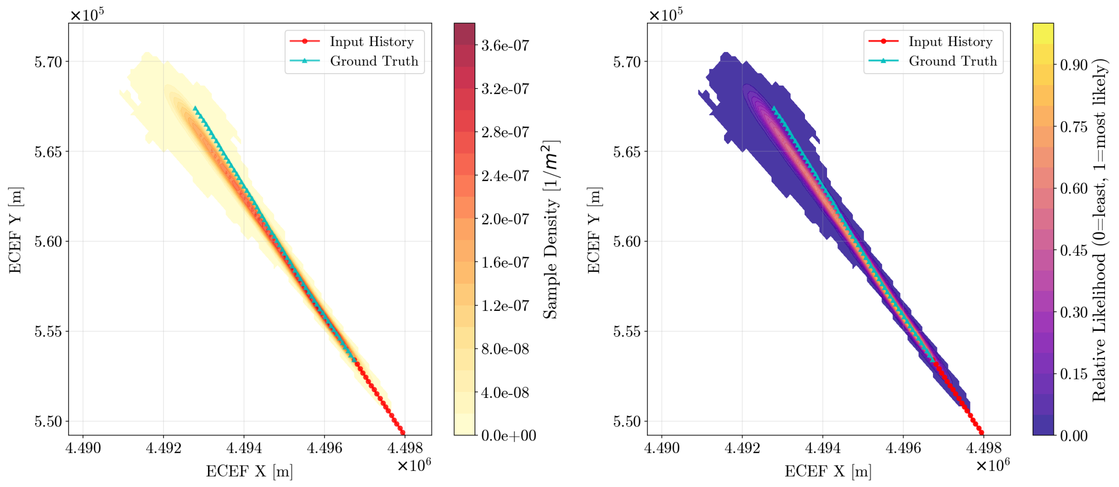


FIGURE 3.5: Density and relative likelihood map from 12,200 MDN samples in the X-Y plane for Sample #1199302. The left panel shows the raw sample density, while the right panel displays the normalized relative likelihood, emphasizing regions of highest probability.

Figure 3.5 provides a 2D visualization of the predicted spatial distribution for a given sample. A dense core region is clearly visible, with the distribution's spread gradually increasing outward. The highest-density region, representing the most probable trajectory, follows a linear path across the forecast horizon, while outermost areas contain fewer samples and correspond to lower predicted likelihood. To further examine the temporal evolution of this spatial distribution, Figure 3.6 presents three representative prediction horizons for the same test sample.

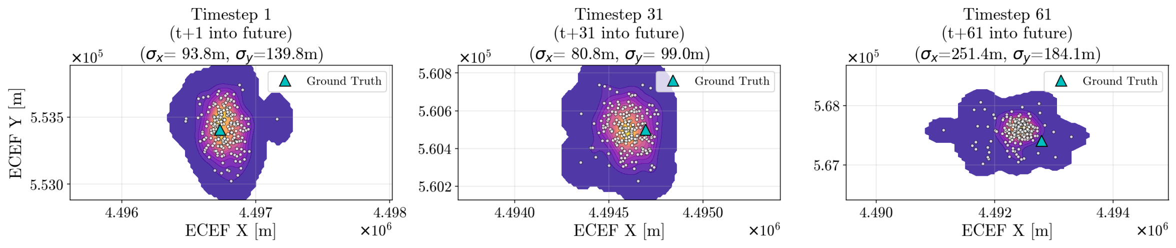


FIGURE 3.6: Predicted spatial distributions and contour plots of the MDN output in the X-Y plane at prediction horizons of 1, 31, and 61 seconds for Sample #1199302. White dots represent samples drawn from the predicted distribution, and the cyan triangle denotes the ground truth. σ_X and σ_Y indicate the standard deviations along each axis at each timestep.

Figure 3.6 clearly shows that the forecasted distributions broaden over time. At Timestep 1, the distribution appears tightly clustered, with $\sigma_X = 93.8$ m and $\sigma_Y = 139.8$ m. Notably, at Timestep 31, the spread reduces to $\sigma_X = 80.8$ m and $\sigma_Y = 99.0$ m, resulting in a more concise contour envelope and suggesting a temporary refinement of uncertainty for this sample. Conversely, by Timestep 61, the distribution expands significantly, with a maximum observed spread of $\sigma_X = 251.4$ m and $\sigma_Y = 184.1$ m, demonstrating the increasing uncertainty over longer horizons. In each snapshot, the ground truth (cyan triangle) consistently resides within a high-density region of the predicted distribution, with its position at Timestep 61 being the furthest from the central concentration (most uncertain), aligning well with the overall distribution of samples (white dots).

This effect can also be assessed in aggregate by examining how the predicted standard deviations evolve across all test samples and mixture components, as shown in Figure 3.7.

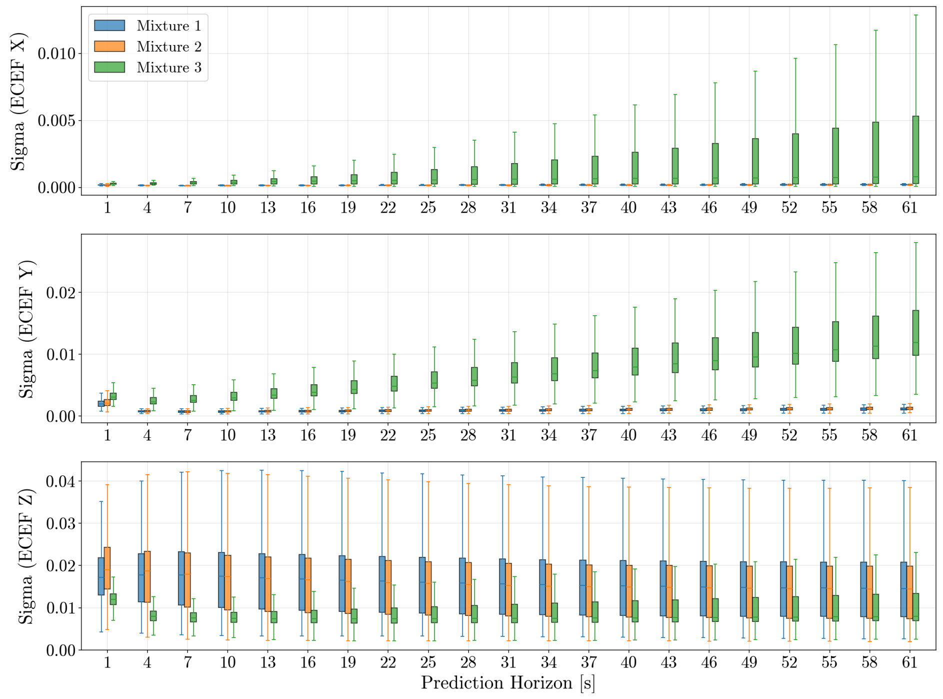


FIGURE 3.7: Box plots of the predicted standard deviations (σ) for X, Y, and Z coordinates across the prediction horizon, grouped by mixture component.

A clear functional specialization among the mixture components is evident from Figure 3.7. Mixture 3 (green) consistently models significantly higher σ values in the horizontal plane (X and Y coordinates), with its median σ values showing a steady increase with prediction horizon, indicating a primary role in capturing growing uncertainty in these dimensions. Conversely, Mixtures 1 (blue) and 2 (orange) predominantly model greater standard deviations along the vertical axis (Z coordinate), where they exhibit notably larger σ ranges compared to Mixture 3, despite their X and Y values remaining small. The increasing spread of the boxes and whiskers for all mixtures with increasing prediction horizon across all dimensions further confirms the general trend of growing prediction uncertainty over time. These observations have been quantified and summarized in Table 3.1.

	Metric	**X**	**Y**	**Z**
Uncertainty Specialization	Sigma Ratio $\sigma_{\max}/\sigma_{\min}$	9.81	8.40	2.87
Spatial Diversity	Mean Prediction Similarity	0.799	0.771	0.684
Global Characteristics	Avg. Mixture Weight (π_k)	**1:** 0.892, **2:** 0.096, **3:** 0.012		
	Mean Weight Entropy	0.363 (33% of max)		

TABLE 3.1: Quantitative analysis of the mixture behaviour.

Table 3.1 reveals a significant weight imbalance, with Mixture 1 exhibiting a dominant average weight ($\pi_k = 0.892$), further supported by the low mean weight entropy (0.363), which corroborates the non-uniform usage of components. The functional specialization, particularly in uncertainty, is quantified by the high Sigma Ratios ($\sigma_{\max}/\sigma_{\min}$) for X (9.81) and Y (8.40), confirming that one component's modelled uncertainty is indeed nearly an order of magnitude larger in the horizontal plane. Despite this specialization and weight imbalance, the mean prediction similarity scores remain well below 1.0, which signifies that the components still generate spatially distinct mean trajectories.

4. Discussion

This section critically evaluates the performance and behaviour of the Bidirectional Long Short-Term Memory (BLSTM) encoder-decoder model with a Mixture Density Network (MDN) for short-term probabilistic aircraft trajectory prediction in an en-route context. The analysis will interpret the quantitative and qualitative results from Section 3, contextualize them within the existing literature, and discuss the model's inherent limitations.

The primary objective of this research was to develop a model capable of quantifying the uncertainty in short-term trajectory predictions for en-route flights. The results affirm that the BLSTM-MDN architecture successfully achieves this fundamental goal. As demonstrated in Figure 3.3, the model generates probabilistic forecasts where the spread of sampled trajectories increases with the prediction horizon, reflecting growing uncertainty over time. Crucially, the ground truth trajectories consistently fall within the high-probability regions of the predicted distributions, validating the model's ability to produce probabilistically meaningful forecasts. However, the investigation also revealed two significant and unexpected behaviours that warrant detailed discussion: a pronounced specialization of the mixture components and an anomalous error spike at the initial prediction timestep.

A central and counter-intuitive outcome was the model's tendency towards mixture component specialization rather than learning true multimodal trajectories. The MDN was designed with three mixture components (K=3), hypothetically to capture distinct future paths (e.g., straight, turning left, turning right). Instead, the model overwhelmingly favoured a single component, which commanded a dominant average weight of 0.892. The resulting low mean weight entropy of 0.363, which is only 33% of the theoretical maximum for three components, confirms a significant lack of probabilistic diversity, a phenomenon often described as mixture collapse (Makansi et al. 2019). This issue is a known challenge in MDNs, often stemming from optimization difficulties or numerical instability. In this case, the components did not collapse into redundancy but rather repurposed themselves to model anisotropic uncertainty. As evidenced in Figure 3.7, the components developed distinct roles: Mixture 3 specialized in modelling the growing uncertainty in the horizontal plane (X and Y axes), while Mixtures 1 and 2 focused on the greater, yet more stable, uncertainty along the vertical axis (Z axis). This behaviour is a direct consequence of the training data's characteristics. The dataset, by design, consisted of en-route segments above FL195, a phase of flight dominated by straight, stable trajectories with minimal manoeuvring. The model, therefore, had little incentive to learn distinct alternative routes and instead found it more efficient to model the different dimensions of uncertainty around a single, highly probable path.

This finding contrasts with other applications of BLSTM-MDN models. For instance, in the maritime prediction done by Sørensen et al. (2022), a similar architecture with more components (K=11) successfully modelled distinct routing choices, such as turning into a harbor versus continuing straight. The limited number of components in this study, combined with the lack of significant manoeuvrer variations in the en-route data, is the most likely cause for this specialization. Furthermore, the loss function, which included a weighted Mean Squared Error (MSE) term (λ = 0.2), may have inadvertently anchored the optimization to a single, accurate mean trajectory, discouraging the exploration of multiple distinct modes.

Another unexpected finding was the anomalous spike in prediction error at the very first timestep seen in Figure 3.2, which then promptly decreased before resuming the expected upward trend over the horizon. This was especially pronounced in the Y-direction, where the mean absolute error at the first step was nearly 83 m before dropping. This anomaly likely stems from the architectural "hand-off" between the encoder and decoder. The encoder processes a 24-second input history, and its final hidden state is used to initialize the decoder to begin generating the output sequence. This transition from consuming known data to generating new data can create a momentary instability as the decoder's internal state settles. Although a one-second overlap was implemented between the input and output sequences to smooth this transition, the results suggest it was not entirely effective. The initial decoder step, even when predicting a point it has technically seen, operates under a different mechanism (generation vs. encoding) and struggles to perfectly align its initial state, leading to a discontinuity. This highlights a subtle but important challenge in designing encoder-decoder architectures for continuous, high-precision forecasting.

The model, while demonstrating probabilistic capabilities, has several limitations that frame the results as a foundational baseline rather than an operationally viable solution.

- **Predictive Accuracy:** The model's deterministic accuracy is not competitive with the state-of-the-art. At the 60-second horizon, mean absolute errors reached approximately 194 m in altitude (Z) and 225 m laterally (Y). In contrast, other recent work using BLSTM models, such as the one developed by Yang et al. (2023), for 60-second en-route prediction has reported an altitude Mean Absolute Error (MAE) as low as 8.45 m. This discrepancy underscores that while our model captures uncertainty, its core predictive precision needs significant improvement.

- **Prediction of Absolute vs. Differential Coordinates:** The model was designed to predict absolute future coordinates (latitude, longitude, altitude). Although derived dynamic features such as heading rate and groundspeed change were used as inputs, literature suggests that models predicting differential changes (e.g., change in position) instead of absolute positions often achieve order-of-magnitude performance improvements (Zeng et al. 2022).

- **Absence of External Contextual Data:** The model operates in an information vacuum, lacking crucial external data known to influence flight paths. Trajectory deviations are often driven by meteorological factors like wind and convective weather. State-of-the-art approaches by Liu and Hansen (2018) and Pang et al. (2019) explicitly integrate such data, for instance, by using embedded Convolutional Neural Networks (CNNs) to extract relevant weather patterns. The absence of these inputs fundamentally limits the model's predictive ceiling, as it can only learn from historical trajectory shapes without understanding their underlying causes.

- **Homogeneity of Training Data:** The focus on en-route segments above FL195 created a dataset with limited dynamic diversity. Research has consistently shown that training distinct models for specific flight phases or manoeuvrers (e.g., turns, level-offs) yields higher accuracy (Le et al. 2020). The observed mixture component specialization is a direct consequence of this limited manoeuvrer diversity; the model had no incentive to develop distinct probabilistic modes because there were few distinct futures to learn from in the data. Future work should explore training specialized models or using a more varied dataset that includes transitions between flight phases.

5. Conclusion and Outlook

This research set out to investigate the application of a Bidirectional Long Short-Term Memory (BLSTM) architecture with a Mixture Density Network (MDN) for probabilistic short-term aircraft trajectory prediction in en-route airspace. The central objective was to move beyond deterministic forecasts and develop a model capable of quantifying the inherent uncertainty in flight paths, a critical requirement for enhancing safety in increasingly flexible environments like Free Route Airspace (FRA).

The primary research objectives have been met. A model was successfully developed and trained on historical ADS-B data, and it demonstrated the fundamental capability to generate probabilistic forecasts. The results confirmed that the model's predicted uncertainty, represented by the spread of sampled trajectories, realistically increased with the prediction horizon, at least from beyond 7-9 seconds. Furthermore, the ground truth trajectories were consistently enveloped within the high-probability regions of the predicted distributions, validating the model's capacity for meaningful uncertainty quantification.

However, interpreting the findings at a higher level of abstraction reveals a crucial insight into the nature of trajectory uncertainty in the en-route phase. The model's mixture components did not learn distinct, multimodal flight paths as hypothesized. Instead, they evolved to perform a more nuanced task: modelling anisotropic uncertainty, with different components specializing in capturing the magnitude of error along horizontal and vertical axes. This suggests that for the stable, linear-dominant trajectories characteristic of en-route flight, the model found it more effective to describe the shape of the uncertainty around a single probable path rather than to predict discrete alternative routes.

While successful in its primary goal of probabilistic forecasting, the research also highlights the model's significant limitations for operational use. The achieved deterministic accuracy falls short of state-of-the-art benchmarks, indicating that the model's practical utility for high-precision tasks is currently limited. The findings therefore present a foundational proof-of-concept, demonstrating both the potential and the inherent challenges of applying MDN-based architectures to this specific domain.

The findings and limitations of this study directly inform several promising directions for future research, which are not merely suggestions for different approaches but are targeted responses to the specific challenges uncovered by the current work.

(1) Enhancing Core Predictive Accuracy: The model's suboptimal accuracy must be addressed. A primary avenue for improvement lies in reframing the prediction task from forecasting absolute coordinates to predicting differential states (e.g., changes in position and velocity). This approach often yields more stable and accurate results by focusing the model on learning the system's dynamics. Furthermore, the integration of crucial external context, particularly meteorological data such as wind fields and temperature, is essential for moving beyond purely kinematic extrapolation and capturing the environmental forces that influence trajectories.

(2) Stochastic Modelling with Bayesian Networks: The observed mixture component specialization reveals a limitation of using MDNs for this context, since the model quantifies uncertainty implicitly without representing well its underlying causes. A compelling and sophisticated next step is to explore more structured probabilistic graphical models, such as Bayesian Networks. A Bayesian framework would allow for the explicit modelling of the causal relationships and conditional dependencies between an aircraft's state and the factors driving deviations, such as proximate weather, Air Traffic Control (ATC) instructions, or sector congestion. This would represent a paradigm shift from quantifying what the uncertainty is to modelling why it exists, leading to more robust, interpretable, and trustworthy probabilistic forecasts essential for safety-critical applications like Conflict Risk Management (CRM). This approach directly addresses the need for a more granular understanding of uncertainty that the current model could not provide.

(3) **Improving Model Generalization and Specialization:** The homogeneity of the en-route dataset was a key factor in the model's behaviour. Future work should focus on developing more generalized models by training on a more diverse dataset that includes different flight phases (climb, descent, approach) and manoeuvrers (turns, holding patterns). This could involve training phase-specific expert models or using transfer learning techniques to adapt a base model to different flight regimes. This would not only improve overall accuracy but also provide a richer environment to test whether a model can learn true multimodality when presented with more complex and varied scenarios.

In summary, while this thesis has successfully demonstrated a method for probabilistic trajectory prediction, its most valuable contribution may be the clear signposting of the path forward. Future research should focus on integrating external causal factors and adopting more structured probabilistic methods to build the next generation of trajectory prediction tools that are not only accurate but also robustly and transparently account for the complex uncertainties of real world flight operations.

A. Python Code

This section provides a concise overview of the `VT_1` project's core Python scripts and Jupyter notebooks, illustrating their role within the comprehensive data processing and machine learning pipeline. The codebase is designed for both clarity and modularity, guiding users from raw data ingestion to model inference. The complete repository is accessible on GitHub: https://github.com/alexfustagueras/VT1.

The `VT_1` directory is structured as follows:

```
VT_1/
├──01_preprocessing.ipynb
├──02a_filtering.ipynb
├──02b_transforming.ipynb
├──02c_generating_samples.ipynb
├──03_model_training.py
├──04_inference.ipynb
├──traffic.yml
├──tf-mdn.yml
└──README.md
```

The project workflow is organized into distinct phases, beginning with data preparation and proceeding through model development, supported by well-defined environments.

- **Data Preparation Notebooks:**
  - `01_preprocessing.ipynb`: Handles the initial loading and filtering of raw data.
  - `02a_filtering.ipynb`: Focuses on extracting and cleaning relevant en-route trajectories.
  - `02b_transforming.ipynb`: Performs feature engineering and transforms data into a suitable format for modeling.
  - `02c_generating_samples.ipynb`: Creates the necessary input and output sample pairs for model training.
- **Model Development:**
  - `03_model_training.py`: This is the standalone script for robust model training and evaluation.
  - `04_inference.ipynb`: Demonstrates the model's capabilities by performing inference and visualizing prediction uncertainties.
- **Environment and Project Setup:**
  - `traffic.yml`: Defines the Conda environment for data preprocessing, including the `traffic` library.
  - `tf-mdn.yml`: Manages the Conda environment for model development and training, particularly for TensorFlow with Mixture Density Networks.

Bibliography

Ayala, D., Ayala, R., Vidal, L. S., Hernández, I., & Ruiz, D. (2023). Neural networks for aircraft trajectory prediction: Answering open questions about their performance. *IEEE Access*, *11*, 26593–26610. https://doi.org/10.1109/ACCESS.2023.3255404

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. https://doi.org/10.1145/2939672.2939694

Brown, M., Lee, K., Kim, H. Y., Hirabayashi, H., Murata, A., Kim, H., Park, S. E., & Gray, N. H. (2024). Introducing free route airspace to northeast asia/pacific region. *Transactions of the Japan Society for Aeronautical and Space Sciences*, *67*(6), 350–361. https://doi.org/10.2322/tjsass.67.350

Çeçen, R. K., & Çetek, C. (2019). A two-step approach for airborne delay minimization using pretactical conflict resolution in free-route airspace. *Journal of Advanced Transportation*, *2019*, 1–17. https://doi.org/10.1155/2019/4805613

EUROCONTROL. (2025a). *Eurocontrol seven-year forecast 2025-2031* (Technical Report). EUROCONTROL. https://www.eurocontrol.int/publication/eurocontrol-forecast-2025-2031

EUROCONTROL. (2025b). *Free route airspace (fra) implementation projection charts*. EUROCONTROL. https://www.eurocontrol.int/publication/free-route-airspace-fra-implementation-projection-charts

European Parliament and Council. (2011). Implementing regulation - 2021/116. https://eur-lex.europa.eu/eli/reg_impl/2021/116/oj

FABEC. (2022). Free route airspace adds swiss entire upper airspace, adjoining french airspace, and german cross-border routes. https://www.atc-network.com/atc-news/fabec/fabec-free-route-airspace-adds-swiss-entire-upper-airspace-adjoining-french-airspace-and-german-cross-border-routes

Garcia-Chico, J., Vivona, R., & Cate, K. (2008). Characterizing intent maneuvers from operational data: Step towards trajectory prediction uncertainty estimation. In *Aiaa guidance, navigation and control conference and exhibit*. https://doi.org/10.2514/6.2008-6520

International Civil Aviation Organization. (2005). Global Air Traffic Management Operational Concept (First).

Le, T. H., Tran, N. P., Pham, D.-T., Schultz, M., & Alam, S. (2020). Short-term trajectory prediction using generative machine learning methods.

Liu, Y., & Hansen, M. (2018). Predicting aircraft trajectories: A deep generative convolutional recurrent neural networks approach. https://arxiv.org/abs/1812.11670

Makansi, O., Ilg, E., Cicek, O., & Brox, T. (2019). Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7137–7146. https://doi.org/10.1109/cvpr.2019.00731

Nava Gaxiola, C. A., Barrado, C., Royo, P., & Pastor, E. (2018). Assessment of the north european free route airspace deployment. *Journal of Air Transport Management*, *73*, 113–119. https://doi.org/https://doi.org/10.1016/j.jairtraman.2018.08.008

Olive, X. (2019). Traffic, a toolbox for processing and analysing air traffic data. *Journal of Open Source Software*, *4*, 1518. https://doi.org/10.21105/joss.01518

Pang, Y., Xu, N., & Liu, Y. (2019). Aircraft trajectory prediction using LSTM neural network with embedded convolutional layer. *Annual Conference of the PHM Society*, *11*. https://doi.org/10.36001/phmconf.2019.v11i1.849

Rocha Murça, M. C., & 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. https://doi.org/10.1109/DASC50938.2020.9256644

Schäfer, M., Strohmeier, M., Lenders, V., Martinovic, I., & Wilhelm, M. (2014). Bringing up opensky: A large-scale ADS-B sensor network for research. *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, 83–94. https://doi.org/10.1109/IPSN.2014.6846743

SKYbrary Aviation Safety. (2025). Free route airspace (FRA). *EUROCONTROL*. Retrieved May 25, 2025, from https://skybrary.aero/articles/free-route-airspace-fra

Sørensen, K. A., Heiselberg, P., & Heiselberg, H. (2022). Probabilistic maritime trajectory prediction in complex scenarios using deep learning. *Sensors*, *22*(5). https://doi.org/10.3390/s22052058

Tamara Pejovic, G. P., Antonio Lazarovski. (2019). Impact of free route airspace implementation on safety performance. ex-post analysis of northern europe free route airspace (nefra). *SESAR Innovation Days 2019*. https://www.sesarju.eu/sites/default/files/documents/sid/2019/papers/SIDs_2019_paper_49.pdf

Tominaga, K., Chan, A. K. W., Sultana, J., Schultz, M., Itoh, E., & Duong, V. N. (2023). Operational feasibility assessment of the free route airspace concept in the asean region. https://www.atrsworld.org/2023

Yang, Z., Kang, X., Gong, Y., & Wang, J. (2023). Aircraft trajectory prediction and aviation safety in ADS-B failure conditions based on neural network. *Scientific Reports*, *13*(1), 19677. https://doi.org/10.1038/s41598-023-46914-2

Zeng, W., Chu, X., Xu, Z., Liu, Y., & Quan, Z. (2022). Aircraft 4d trajectory prediction in civil aviation: A review. *Aerospace*, *9*(2). https://www.mdpi.com/2226-4310/9/2/91

Zhao, Y., Yang, R., Chevalier, G., Shah, R. C., & Romijnders, R. (2018). Applying deep bidirectional LSTM and mixture density network for basketball trajectory prediction. *Optik*, *158*, 266–272. https://doi.org/https://doi.org/10.1016/j.ijleo.2017.12.038