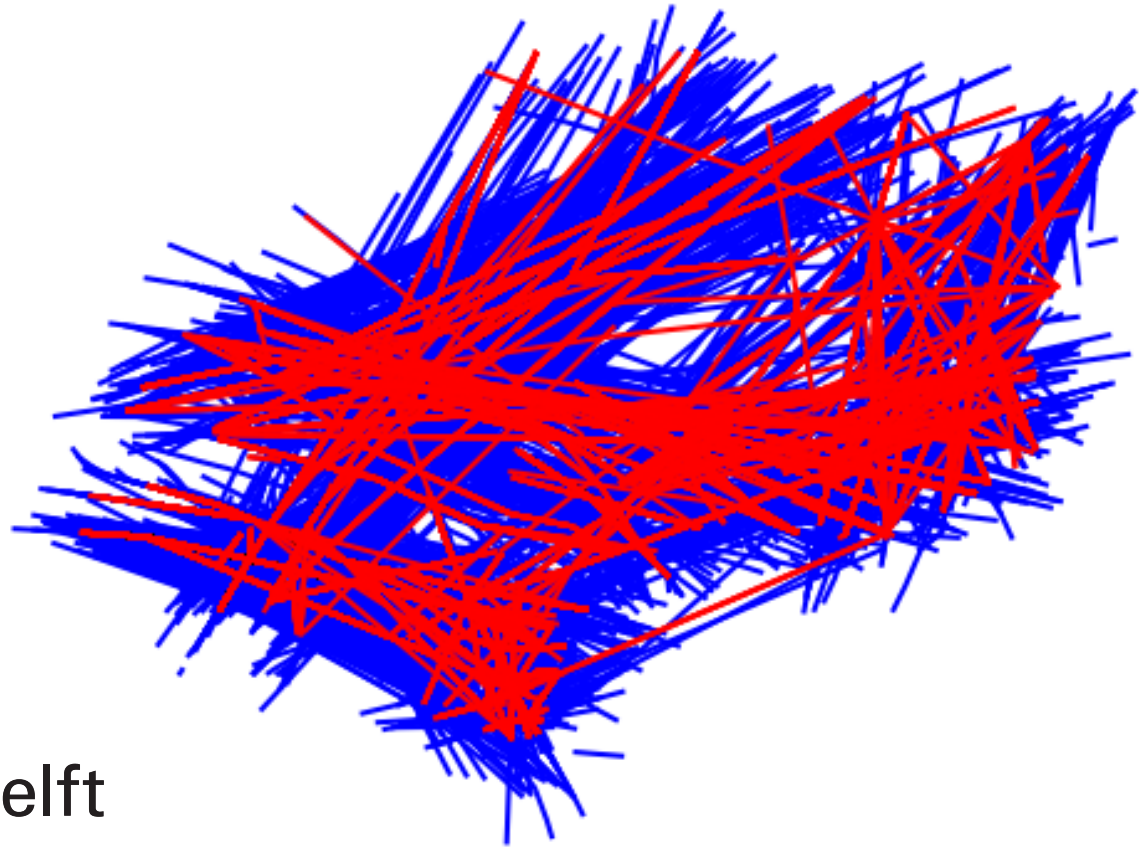
A Machine Learning Approach to Evaluating Aircraft Deviations from Planned Routes

M.K. Sakyi-Gyinae

Department of Control and Simulation
Faculty of Aerospace Engineering



TU Delft

MSc. Thesis Report

A Machine Learning Approach to Evaluating Aircraft Deviations from Planned Routes

by

Master K. Sakyi-Gyinae

Delft University of Technology

Supervisors: PhD Candidate J. Rudnyk C&S Aerospace Engineering TU Delft
Assistant Professor J. Ellerbroek C&S Aerospace Engineering TU Delft
Professor J.M. Hoekstra C&S Aerospace Engineering TU Delft

TU Delft

Preface

The journey to obtain my master's degree at the faculty of aerospace engineering in Delft has exceeded my expectations and appreciation for technology. To be able to conclude through writing this thesis has been a true honor.

Firstly, I would sincerely like to thank my supervisors Jacco Hoekstra, Joost Ellerbroek and Julia Rudnynk for their guidance and feedback. I truly would not have been able to accomplish this without them.

Furthermore, credits is due to Ihor Smal and Junzi Sun for being involved in the discussion of this project. I would also like to offer special thanks to Laura van Vaeck and Ka-kin Fung for their feedback related to the technical writing of this report. Lastly, I would like to thank my family and friends for their unconditional support throughout the whole process.

M.K. Sakyi-Gyinae
Delft, Oktober 2019

Contents

List of Figures

List of Tables

Acronyms

ANN Artificial Neural Network. vii, 11–15, 30

ANSP Aircraft Navigation Service Provider. 28, 50, 53

ATC Air Traffic Control. iii, 1, 2, 4, 5, 7, 28, 34, 50, 51

ATCo Air Traffic Control operator. vii, 1, 2, 5, 6, 25, 26, 29, 34, 35, 39, 51

ATE Along-track Error. v, vii, 6–8, 25, 26, 35, 47

ATFCM Air Traffic Flow and Capacity Management. 3, 50–52

ATFM Air Traffic Flow Management. iii, 4, 30, 50, 51

ATM Air Traffic Management. v, 1, 2, 25, 26, 28, 50, 54

ATS Air Traffic Services. 4

AU Airspace Users. 53

BGD Batch Gradient Descent. 17

BPTT Back Propagation Though Time. v, 18–20

CNS Communication, Navigation and Surveillance. 4

CTE Cross-track Error. v, vii, 6–8, 25, 26, 47–49, 54, 56

DRNN Deep Recurrent Neural Network. 17

DST Decision Support Tool. 1, 2

EOBT Estimated Off-block Time. 4

FFNN Feed Forward Neural Network. 14, 15, 17

FL Flight Level. 34, 55

FPL Filed Flight Plan. v–vii, 1–9, 12, 17, 25, 26, 28, 30–32, 34–36, 39, 46–53, 56

ICAO International Civil Aviation Organization. 4, 5, 26, 35, 39

iFMP integrated Flow Management Position. 4, 50

IFR Instrument Flight Rules. 1, 35

LSTM Long Short Term Memory. v–vii, 19–21, 26, 42, 44–53, 56, 57

MBGD Mini-Batch Gradient Descent. 16, 17

ML Machine Learning. 26

MUA Maastricht Upper Airspace. vi, vii, 26, 27, 39, 40, 46–49, 51, 52

MUAC Maastricht Upper Airspace Control. v–vii, 28–30, 34, 36, 37, 39

NM Network Manager. 50, 53

NN Neural Network. 12, 17

PAE Pressure Altitude Error. v, vii, 6–8

PCA Principle Component Analysis. 37, 38, 42

RelU Rectified Linear unit. 14, 42

RF Random Forest. 11

RNN Recurrent Neural Network. v, 17–20, 25–27

SGD Stochastic Gradient Descent. 17

SVM Support Vector Machine. 11, 12, 15

TP Trajectory Predictor. 1–3, 5, 6, 25, 50, 53, 54

TPE Trajectory Prediction Error. 4, 6, 7, 40, 48, 54

1 Introduction

Recent figures published by Eurocontrol predict an annual 12.4 millions flights by 2024 [8]. This is approximately an increase of 17% more Instrument Flight Rules (IFR) flights than the total 10.6 million flights measured in 2017. The irreversible demand in air traffic creates a need for increased airspace capacity. To accomplish this, the limits of current and future Air Traffic Management (ATM) systems need to be expanded to allow a more efficient use of airspace. This could however be complicated by fragmented airspace such as the current European airspace [9], which results in its inefficient use and complex air traffic operations. Moreover, fixed airways prevent aircraft from flying direct routes, leading to an increased flight time, additional fuel burn and more CO_2 emissions. These constraints could restrict the growth in airspace capacity.

Alongside the technological advancements proposed and developed by research and finance programs such as SESAR[10], more insight is needed into how to increase airspace capacity. The problem of limited capacity is related to the mental workload of Air Traffic Control operators (ATCos) [11], who are responsible for safe and expeditious flow of air traffic [12]. This is accompanied by a highly intensive monitoring task, which limits the number of aircraft an ATCo can safely handle [13]. To overcome this limitation, Decision Support Tools (DSTs) are developed to aid ATCos in their cognitive control task, hence providing the possibility to increase airspace capacity. Many DSTs require predictions of aircraft trajectories, which are not always accurate due to several factors, including inaccurate flight intent data, i.e. the intended path an aircraft is going to follow. Currently, the filed flight plan (FPL) and its amendments are the only source of information constructing the flight intent. The flight route related fields of a flight plan used today date back to the 1960s [11] and a route construction is mainly based on navigation points along fixed airways. In reality, aircraft do not always follow the route specified in a FPL and thus diverge from the reference trajectory. For example, Fig. 1.1 illustrates the route from Schiphol to Munich, where on the left 110 reference trajectories are generated based on the FPL and on the right the actual tracks of these flights are displayed.

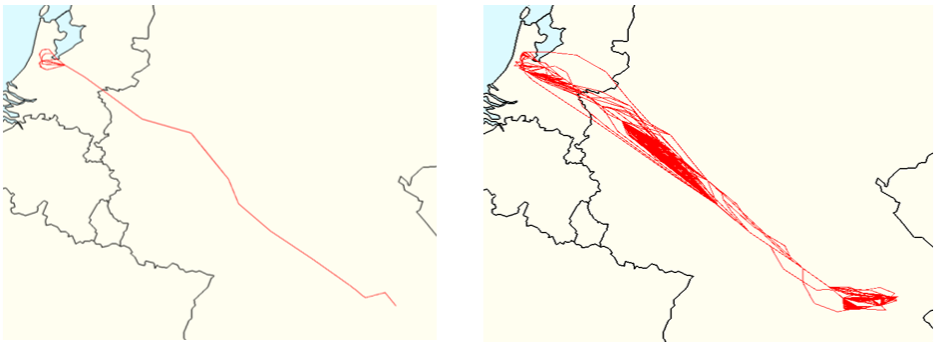


Figure 1.1: Route Schiphol(Amsterdam)-Munich. Left: 110 reference trajectories based on FPL. Right: 110 flown trajectories. The data were obtained from the Demand Data Repository of Eurocontrol [1] and plotted using Eurocontrol SAAM tool [2]

The difference between the reference and flown trajectory is known as deviation and can be decomposed into vertical, longitudinal, and lateral deviations (also respectively called vertical, along-track, and cross-track errors). These deviations can be caused by severe weather conditions, restricted and or congested airspace, clearances issued by Air Traffic Control (ATC) (e.g. route shortcuts or due to conflicting traffic), and pilot-ATCo miscommunication. Furthermore, a difference between the reference and flown trajectory can occur when an issued clearance was not input into the controller's host computer after being issued over voice. Since a Trajectory Predictor (TP) will lack relevant intent data, this will result in inaccurate predictions, which

in turn invalidates the performance of a DST. As a result, the discrepancy between the information provided by a DST and reality will increase the workload of an ATCo and limit the opportunity to increase capacity. By analyzing deviations before they actually occur, the accuracy of TP algorithms and DSTs can be improved for longer look-ahead time [14, 15], which should enable ATCos to plan air traffic better and thus cope with a higher capacity. Subsequently, this would have an effect in upgrading present and future ATM systems.

Much research has been conducted in detecting deviations in air traffic flow, which sometimes is referred to as conformance monitoring. Although conformance monitoring is related to detecting deviations from FPL in flight, there is a slight distinction. Suppose a direct flight is permitted with an issued heading vector clearance. Although the aircraft has deviated from the FPL, it can be still conforming to the path defined by the heading vector. Reynolds et al. (2005) used fault detection techniques to create a conformance monitoring algorithm. In this approach a decision on (non-)conformance was based on conformance residuals, being the difference between observed and expected states [16]. Employing a similar technique, algorithms created by Fourlas et al. (2006, 2007) could make a distinction in detecting non-conformance due to nominal (wind) and non-nominal uncertainties (e.g. ATC and pilots induced errors) in the horizontal and vertical plane [17, 18]. Both approaches used state residuals and predefined thresholds. In other studies [19, 20], conformance monitoring frameworks capable of handling uncertainties were introduced by employing conformance probability.

Due to the increase in available ATM data, processing and creating models with a large-scale data set is becoming more of a challenge. Deterministic algorithms cannot cope as they are limited in efficiency, robustness and scalability [21]. However, evaluation of a population is better reflected with a bigger sample size. Consequently, together with its predictive capabilities and complex algorithms, machine learning has been adopted in ATM as a big data mining technique to overcome these issues.

In previous research, supervised and unsupervised machine learning have been used to identify flight trajectory patterns in historical data sets. Sun et al. (2016) used an unsupervised clustering method, density-based spatial clustering of applications with noise (DBSCAN), to cluster large historical volume of flights transmitting ADS-B data into individual flights [21]. Fuzzy logic approach was implemented to further label the output of the initial clusters into flight phases. Fuzzy theory sets have been widely implemented in machine learning [22] and can be thought of as a good candidate to identify deviated segments in air traffic flow.

Similarly, Murca et al. (2016) proposed a framework that employs unsupervised and supervised learning algorithms applied to flown trajectories to detect airspace operational patterns, classify trajectories, and to identify non-conforming flights in the transition airspace (arrival and departure) [23]. In yet another study, a clustering algorithm was proposed to monitor the airspace and assess flight deviations [24]. Finally, Naessens et al. (2018) used deep neural networks to predict a subset of routes, starting from UK to European destinations to the south and southeast [25].

Few studies have been performed to objectively identify the causes of deviations. In the Eurocontrol Flight Data Management Metrics project (2006), of which the results were restated in [15](2009), minor (between 20 and 50 nautical miles) and major (more than 50 nautical miles) categories were used to classify lateral deviations. According to that project, 19% of 27,300 flights evaluated in the European airspace deviate laterally with an an average of 30 nautical miles, and about 3% deviate up to 73 nautical miles on average. The cause for these deviations was characterized as direct routing to downstream waypoints. When assessing lateral deviations that occur in the United States airspace by means of hierarchal clustering, it was inferred that major deviations were caused due to convective weather and airspace structures with increased number of turns [15]. More recently, Bongiorno et al. (2017) evaluated the relationship between airspace structure and deviations by introducing the di-fork metric, which allowed to identify navigation points where deviations were more or less frequent to occur [6].

Although several studies proposed methods to detect and predict deviated routes and infer their cause using statistical and machine learning techniques for transitional phases of the flight [6, 15, 19, 23–25], they did not take into account all sources of deviations. While some focused on detecting non-conformance due to weather, others focused on airspace structure influencing air traffic flows. There are many more contributing factors causing trajectory deviations that should be introduced in the analysis when possible. Moreover, less

attention has been giving to the analysis of deviations experienced en-route. Transition airspace has been more attractive to researchers due to high density air traffic and various arrival and departure patterns. However, with increased air traffic demand, the use of en-route airspace needs to be optimized as well. Gong et al. (2004) proved the importance of detecting anomalies in TP algorithms using probability distributions of the deviations [14]. Detected anomalies revealed trajectory patterns which TP software did not take into account. Despite this interest, the quantification of deviations has been overlooked due to the tedious process of obtaining, pre-processing and analyzing large-scale reference trajectories. However, with the advent of improved mainstream computing hardware and era of advanced machine learning algorithms, performing such tasks becomes feasible. Lastly, research aiming to propose methods for predicting trajectories do not directly take into consideration all contributing factors of deviations as input to the models [25–28].

With this in mind, the goal of this study is to predict trajectories while taking into account the contributing factors of deviations. This will be done by applying machine learning techniques to real historical trajectory data and FPL data obtained from Maastricht Upper Area Control (MUAC). Furthermore, analysis and quantification of the deviations w.r.t. the trajectories will be performed. With newly gained insights, the quality of input data to TPs can be increased and improvements to ATFCM systems can be achieved. To accomplish this, a pre-analysis of deviations is presented in Chapter 2. This is followed by the time series characteristics of aircraft trajectories in Chapter 3. A review of the methodology is presented in Chapter 4. In Chapter 5 the research design is described. Here, an overview of the objective including the research questions are presented. Chapter 6 presents the data acquisition and management tools used in this study, followed by the model input selection procedure, model output transformation and trajectory prediction model design. The report ends with the simulation results, post- analysis, conclusions and recommendations in Chapters 7, 8, 9 and 10 respectively.

2
Deviation Pre-Analysis

This Chapter begins with understanding flight planning and how it is utilized in air traffic flow management (ATFM) and ATC. Furthermore, a discussion is given about the types of deviations w.r.t. the planned trajectory, followed by possible contributing factors. Next, metrics are defined to quantify deviations numerically. These metrics are then employed in generating statistics on flights, flown between February 2016 and December 2016, to assess the severity of deviations.

2.1. Flight Plan in ATFM and ATC

The International Civil Aircraft Organization (ICAO) defines a flight plan as specified information that is provided to air traffic services along the intended flight or portion of flight an aircraft is travelling [11]. Before take-off, each aircraft operator or pilot submit a flight plan to the network manager (NM) or to air traffic navigation services en-route. In a practical sense, prior to departure, the flight plan must not be submitted more than 5 days or, equivalently 120 hours before the estimated off-block time (EOBT) and at least 3 hours before EOBT. In case the controlled flight is delayed for 30 minutes in excess of EOBT, the FPL will be amended or, in the worst case, a new flight plan has to be submitted whereby the prior flight plan is cancelled [11]. Once submitted and accepted without any subsequent changes, it is known as a FPL. ICAO introduced the flight plan's first form (AIREP and POMAR) in 1960. Since then, it has been updated multiple times, with the most recent update in 2012, where changes have been made such as new indicators for communication, navigation and surveillance (CNS) equipment and capabilities. A flight plan contains fields for aircraft and airport information, route, navigation aids (navaids) to be used during flight, etc. It is essentially the first source of flight information for pilots, air traffic services (ATS) units and NM in order to complete a flight. ATC uses FPLs to perform surveillance on aircraft in its airspace, while ATFM uses FPLs in its tactical operations to balance demand and airspace capacity. An example of an application using the FPL for ATFM is the Integrated Flow Management Position (iFMP) by Eurocontrol [29]. It is a new technological development tool for making initial flow management decisions. It predicts 6 hours in advance the traffic situation within a sector based on entry rates and active local aircraft. The optimizer then identifies underload or overload sector capacity situations. One of its input are trajectory predictions which, when improved, will improve the accuracy of the iFMP.

2.2. Trajectory Prediction Error

Trajectory prediction errors (TPEs) are often used for assessing non-conformance between a real trajectory and a predicted reference trajectory. Three TPEs are commonly used to distinguish different types of deviations: cross-track error, along-track error and altitude error [30].
Cross-track error (CTE) is defined as the perpendicular distance measured from the reference course, whereas along-track error (ATE) is the difference in distance between the reference position and actual position, measured parallel along the predicted course as seen in Fig. 2.1. Lastly, altitude error or pressure altitude error (PAE) is the difference between the predicted and true altitude, depicted in Fig. 2.2.

2.3. Influencing Factors

Due to navigation, flight technical and sensing errors, an aircraft can follow its planned trajectory to a certain degree. It will be bound to deviate from this path within a containment region. This primary type of deviation behaviour is also referred to as *inherent deviation* in this report. The containment region is therefore defined as the region around the nominal path that permits inherent deviations. Fig. 2.3 displays a 2-dimensional containment region where margins are determined by thresholds. The value of the thresholds however do

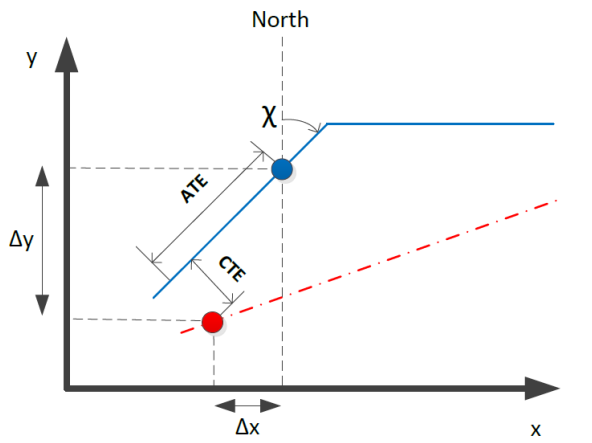


Figure 2.1: Horizontal deviations (ATE and CTE) between the flown trajectory (red) and FPL route (blue)

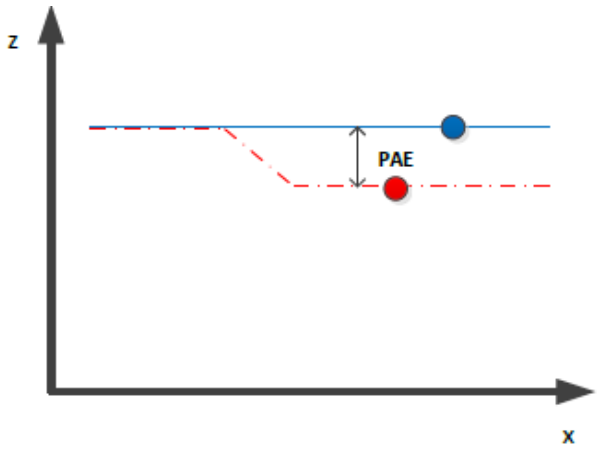


Figure 2.2: Vertical deviations (PAE) between the flown trajectory (red) and FPL route (blue)

not necessarily have to be the same as in required navigation performance, in which an aircraft is required to have a specific navigation performance in order to operate within a certain airspace [19]. As soon as the aircraft leaves the containment region, it is considered as deviating from its FPL, which will be denoted as *secondary deviation* in this report. ICAO (2016) states that a flight is prohibited to deviate from FPL unless the appropriate clearance is given by an ATCo[11]. This means that secondary deviations occur due to a request from the pilot or a control instruction given by an ATCo. In order to connect various factors causing the aircraft to deviate, the interaction between pilot and ATCo should be evaluated when possible.

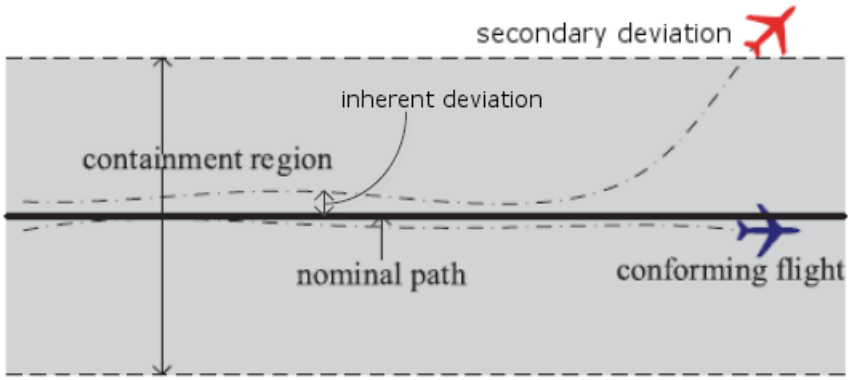


Figure 2.3: A containment region in which the both aircraft deviate inherently from the nominal path but the red aircraft performs a secondary deviation by leaving the containment region

Mondoloni et al.(2006) performed a study on TP inaccuracies and their causes such as variability in aircraft mass, flight management system settings and omission of modelling approaches in relevant flight phases, i.e. turns, holding patterns, etc.[31]. While the focus of much research lies in improving the performance of TPs by using more complex modelling approaches, this study seeks to find out why aircraft deviate from their FPLs. That knowledge can then be used to improve the predictability of TPs in a data-driven way.

Fourlas et al. (2007) classified sources of deviations into 2 groups, the nominal and non-nominal uncertainties [18]. The former being generic uncertainties due to delays, wind, etc., whereas the latter consisted of extreme weather, ATC error (e.g. giving the wrong command, or giving a command to the wrong aircraft), etc. Based on literature review[6, 15, 17, 18], the most significant sources of secondary deviations:

- Atmospheric Conditions
- Convective Weather
- Change in Economic Route
- Miscommunication between an ATCo and a pilot
- Conflicting Air Traffic
- Complex Air Traffic Conditions
- Availability of Restricted Airspace
- Airspace Structure

2.4. Quantifying Deviations

To compute deviations, a time-dependent and waypoint-independent similarity metric is employed. It uses the Euclidean distance as average cost to compare 2 similar trajectories. Therefore, the geodetic coordinates (latitude, ϕ, longitude, λ and height h) of trajectories are transformed to Cartesian coordinates (x, y, z) using Eqs. 2.1 to 2.3, where e is the eccentricity and N is the radius of curvature in the prime vertical of an ellipsoidal model of the earth. h is the height above the earth surface depicted in the earth-centered, earth-fixed (ECEF) coordinate system presented in Fig. 2.4. By using the Cartesian coordinate system, the similarity metric does not have to account for the negligible effect of curvature of the earth. Furthermore, having the origin of the coordinate system at the center of the earth $(0,0,0)$ reduces the complexity of comparing similar trajectories. On the contrary, a local reference frame such as the north-east-down (NED) system which is also used in aircraft navigation, finds its origin fixed at the aircraft's center of gravity. Therefore, an additional transformation step would be required to estimate the similarities between planned and flown trajectories.

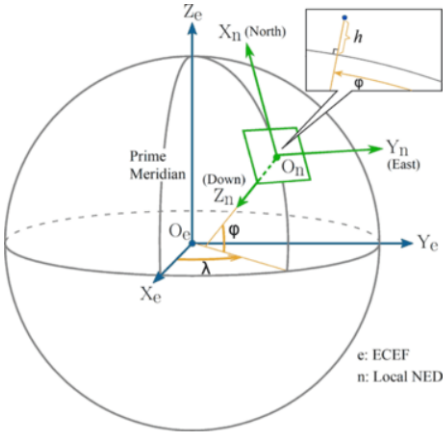


Figure 2.4: Geodetic, ECEF and local NED Reference Frame [3]

$$x = (N+h)\cdot\cos\phi\cdot\cos\lambda \tag{2.1}$$

$$y = (N+h)\cdot\cos\phi\cdot\sin\lambda \tag{2.2}$$

$$z = N(1-e^2)+h\cdot\sin\phi \tag{2.3}$$

The TPEs are then found with Eqs. 2.6 to 2.8, where a 2-dimensional rotation matrix is used for calculating ATE and CTE. ΔX and ΔY, depicted in Fig. 2.1 and defined by Eqs. 2.4 and 2.5, are the differences in reference and flown coordinates. In the same figure, χ represents the track angle of the route described by the FPL. Note that in Eq. 2.4 to 2.8, a variable sub-scripted by p relates to the predicted route of the FPL and t to the real trajectory. Furthermore, errors computed in the Cartesian coordinate system will be smaller in value as the curvature of the earth is neglected here. This reduction is irrelevant due to size of the airspace being evaluated. Conventionally, the units of the TP errors are not decisive. Nautical miles for CTE and ATE, and feet for PAE have been used in various research. ATE can also be measured in seconds based on the estimated time of arrival and actual time at a certain position. Furthermore, PAE can be induced by along-track error and vice versa [31]. Consequentially, coupling of all errors can exist in multiple dimensions [31, 32]. Defining deviations for curvilinear paths becomes more complex than for linear paths. A multidimensional coupled error can also be present in curvilinear flight paths. The scenario of an aircraft making a turn with the

different turn radius and altitude than the predicted path demonstrates the modelling inaccuracies, leading to multidimensional coupled deviations.

$$\Delta X = x_p - x_t \tag{2.4}$$
$$\Delta Y = y_p - y_t \tag{2.5}$$
$$ATE = \Delta X sin\chi_p + \Delta Y cos\chi_p \tag{2.6}$$
$$CTE = \Delta X cos\chi_p - \Delta Y sin\chi_p \tag{2.7}$$
$$PAE = alt_p - alt_t \tag{2.8}$$

2.5. Statistics on Trajectory Predictions Errors

The statistics presented in [15] are outdated and restricted to smaller airspace, therefore new insights are needed on the behaviour of aircraft deviations globally. Using the equations from section 2.4, statistics are generated on 1,388,338 flights passing through European Civil Aviation Conference (ECAC) airspace between February and December 2016. The goal is to identify the current severity of aircraft deviations.

The approach taken to accomplish this, is to compute lateral, longitudinal and vertical deviations performed by these flights. Consider that the TPE or deviation along each reference path and phase of flight is likely to vary. Commonly, the FPL routes are less accurate for climb and descent phases. These phases highly depend on speed settings and aircraft take-off mass which complex aircraft trajectory models sometimes do not incorporate.

In estimating the TPEs, the real trajectory is re-sampled w.r.t. the planned trajectory such that both trajectories have the same take-off time. This means that ground delay or early departure for each flight is removed from the real trajectory. The statistics are generated by obtaining the maximum deviation along an entire flight and in each dimension.

Furthermore, each computed TPE is then categorized as inherent, on-track minor, on-track major, off-track minor or off-track major deviation. The lower limits of these categories are based on the minimum separation standards of flights under ATC surveillance systems [11]. Results shown in Tables 2.1 to 2.3 indicate that nearly all flights do not follow their FPL to some extent. The majority of aircraft, 59.3%, display on-track minor lateral deviations, while approximately 32% show on-track major lateral deviations. Similarly, nearly half of the flights, about 51%, display on-track minor longitudinal deviation, while approximately 35% exhibit on-track major longitudinal deviation. A close investigation of the off-track major lateral and longitudinal deviations reveal that large errors are due to flights that presumably performed emergency landing. Furthermore, certain flights changed their destination after being airborne. These flights will be filtered out of the dataset. Notably, oceanic flights tend to have greater CTE and ATE when deviations are induced early on in flight. Furthermore, multidimensional coupling with ATE results in a PAE distribution with large errors. This suggests that the maximum vertical deviation is primarily found during the landing or take-off phases for certain flights, with one of the trajectories still in flight while the other is on the ground. These are some helpful insights which can be used in this study. The final distributions are shown in Figs. 2.5 to 2.7 where the kernel density estimation of all distributions are bimodal.

Table 2.1: Statistics on maximum CTE w.r.t. FPL for scheduled flights passing through ECAC from February to December 2016

Category	**Limits**	**Number of flights**	**% of total flights**
Inherent lateral deviation	$CTE <= 5\ nmi$	72,577	5.2
On-track minor lateral deviation	$5\ nmi < CTE <= 20\ nmi$	823,288	59.3
On-track major lateral deviation	$20\ nmi < CTE <= 50\ nmi$	441,258	31.8
Off-track minor lateral deviation	$50\ nmi < CTE <= 100\ nmi$	41,505	3.0
Off-track major lateral deviation	$CTE > 100\ nmi$	9,710	0.7

Table 2.2: Statistics on maximum ATE w.r.t. FPL for scheduled flights passing through ECAC from February to December 2016

Category	Limits	Number of flights	% of total flights
Inherent longitudinal deviation	$ATE <= 10\ nmi$	167,767	12.1
On-track minor longitudinal deviation	$10\ nmi < ATE <= 25\ nmi$	704,433	50.7
On-track major longitudinal deviation	$25\ nmi < ATE <= 75\ nmi$	488,387	35.2
Off-track minor longitudinal deviation	$75\ nmi < ATE <= 150\ nmi$	24,671	1.8
Off-track major longitudinal deviation	$ATE > 150\ nmi$	3,080	0.2

Table 2.3: Statistics on maximum PAE w.r.t. FPL for scheduled flights passing through ECAC from February to December 2016

Category	Limits	Number of flights	% of total flights
Inherent vertical deviation	$PAE <= 1000 ft$	3,451	0.2
On-track minor vertical deviation	$1000\ ft < PAE <= 5000\ ft$	171,896	12.4
On-track major vertical deviation	$5000\ ft < PAE <= 10000\ ft$	623,360	44.9
Off-track minor vertical deviation	$10000\ ft < PAE <= 20000\ ft$	492,644	35.5
Off-track major vertical deviation	$PAE > 20000\ ft$	96,987	7.0

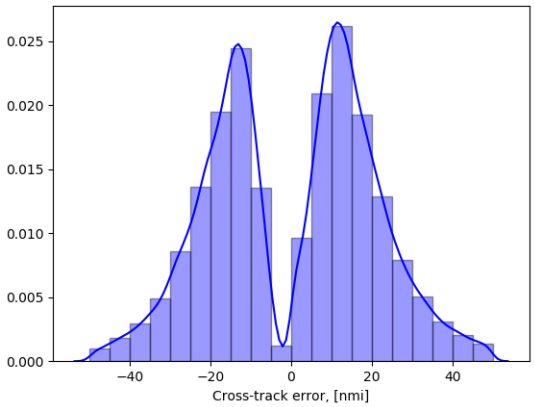


Figure 2.5: Statistics on maximum CTE w.r.t. FPL for scheduled flights passing through ECAC from February to December 2016

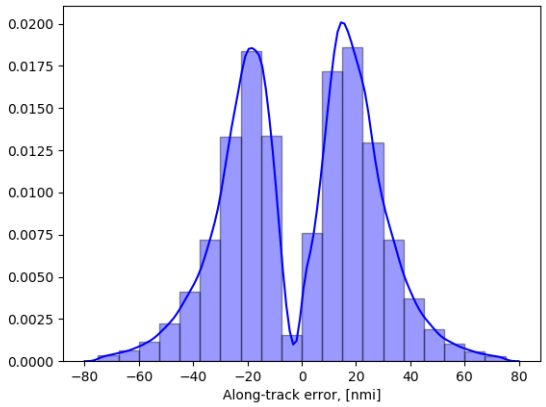


Figure 2.6: Distributin of maximum ATE w.r.t. FPL for scheduled flights passing through ECAC from February to December 2016

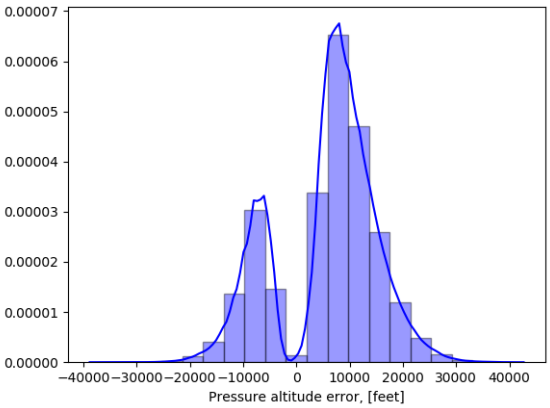


Figure 2.7: Distribution of maximum PAE w.r.t. FPL for scheduled flights passing through ECAC from February to December 2016

3
Time Series Characteristics

The route constructed based on the FPL consists out of a sequence of successive position coordinates (ϕ, γ, h) recorded at each time stamp. Understanding the time series' characteristics of flight trajectories enables the selection of an appropriate model for predicting routes. Therefore, the next subsections evaluate the (non)linear, statistical and observational dependency behaviour of the time series.

3.1. Linear or Nonlinear

According to Philip Chen et al.(2015), there are 2 major categories for predictive models[33]. The first category is the physical approach in which a complex system is solved using correlated data. The other category is based on statistical approaches which use historical data to predict the underlying structure and patterns such as to predict the future. In statistical predictions both linear and nonlinear models are used. Linear models are subject to superposition, meaning that the contribution of each input to the system's output is independent of each other. On the other hand, input that interact in producing the system's output are used in nonlinear models. Although very simple, easy to implement and computationally light, linear models are inaccurate compared to nonlinear models when trying to capture nonlinearities in the data. Contrary to this, nonlinear models such as neural networks are complicated (often using nonlinear optimization algorithms) and computationally expensive. Linear systems are adequate in solving linear problems and therefore reject the need of a nonlinear system for the same problem. Additionally, the presence of nonlinear components as input to a system does not guarantee a nonlinear response. Applying nonlinear models to a linear problem in some cases degrades the performance of the system [33]. It is therefore crucial to distinct the nature of the related problem. By detecting the presence or lack of nonlinearities in the estimation data, a justification can be made on what type of statistical method to use. From the knowledge of individual flights having different entry and exit points within the airspace, it can be assumed that the estimation data contains nonlinearities. This is explained in more detail in chapter 6.

3.2. Short -and Long Range Dependencies

All time series exhibit short -and long range dependencies. This means that a new observation is closely related to past observations separated by either a small or large number of time steps. The degree of the relationship however depends on the type of dataset. The correlations are visible in the time series' (partial) autocorrelation, where persistent autocorrelations suggest a statistical dependency between observations separated by a large number of time units. The opposite is true for short range dependencies, where the autocorrelations tend to rapidly decay [34]. Large dependencies between two consecutive observations can be identified by evaluating the series' partial autocorrelations. The trajectories, analyzed in this report, are non-equidistant time series of which the longitude autocorrelation of a single flight is plotted in Fig. 3.1 on the left. A low order moving average is indicated by the steady decreasing trend in its autocorrelation. On the right side in the same figure, its partial autocorrelation is plotted and suggests a first order autoregressive term. The same trend can be obsevered in the partial and autocorrelation of latitude and pressure altitude. Note that the (partial)autocorrelation magnitudes will be somewhat different for each flight.

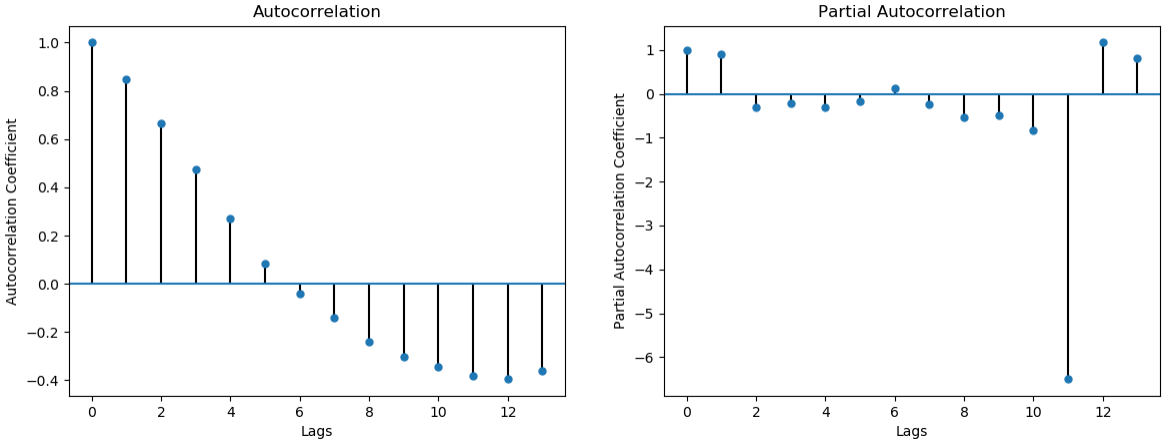


Figure 3.1: Longitude autocorrelation (left) and partial correlation (right) of a single flight where the dots represent correlation magnitude at corresponding lags

4
Methodology

The objective of this study is to develop a model that can predict aircraft trajectories before take-off by using aircraft deviation independent variables. Based on insights obtained from the model, the user can choose which measures to take in order to ensure safe and seamless traffic flow operations. Accomplishing this entails knowing the true coordinates at each time step described by the planned trajectory. In Chapter 3, it is verified that trajectories are nonlinear, multi-dimensional real valued and contain short and long range dependencies. Hence, the prediction of aircraft trajectories from $t = 0$ reduces to a (multi-target) regression problem.

4.1. Regression

Regression is a statistical method that aims to determine the relationship between two or more variables in predictive or casual inference related problems. The statistical method has been adopted in machine learning as a supervised technique which requires sampled sequences, $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$ with corresponding labels or targets $\{y_1, y_2, ..., y_T\}$. The targets act as supervision for an algorithm to learn a mapping function, F, between the input features $\mathbf{x}$ and the targets y. The approximated function, shown in Eq. 4.1, is then used to predict real-valued targets of unseen data.

$$F: \mathbf{x} \mapsto y \tag{4.1}$$

4.1.1. Notation and Problem Formulation

For consistency, the mathematical notation and problem formulation for predicting trajectories is as followed. An n-dimensional exogenous time series, whose values are determined by factors outside F, such as wind and airspace capacity, etc., is given by $\mathbf{X}$, *i.e.*, $\mathbf{X} = (\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^n)^\intercal = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)^\intercal \in \mathbb{R}^{n \times T}$. T is the number of observations or equivalently the length of window. Therefore, $\mathbf{x}^k = (\mathbf{x}_1^k, \mathbf{x}_2^k, ..., \mathbf{x}_t^k) \in \mathbb{R}^T$ is used to represent the $\mathbf{k}^{th}$ series of length T.
A vector of inputs at time t is represented by $\mathbf{x}^k = (\mathbf{x}_t^1, \mathbf{x}_t^2, ..., \mathbf{x}_t^n) \in \mathbb{R}^n$. W.r.t. a multidimensional output, the target series is given by $\mathbf{Y} = (\mathbf{y}^1, \mathbf{y}^2, ..., \mathbf{y}^p)^\intercal = (\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_T)^\intercal \in \mathbb{R}^{p \times T}$. The learning task in Eq. 4.2 is to find a nonlinear mapping function that transforms the exogenous input series to its continuous target series.

$$F: \mathbf{X} \mapsto \mathbf{Y} \tag{4.2}$$

The trained multi-target regression model, F, is then used to predict the values $\left\{\mathbf{y}^{p+1}, ..., \mathbf{y}^{p'}\right\}$ of unseen observations $\left\{\mathbf{x}^{n+1}, ..., \mathbf{x}^{n'}\right\}$.

4.1.2. Motivation Method

Several nonlinear supervised learning techniques exist for regression. These include Bayesian methods, support vector machine (SVM), random forest (RF), Artificial Neural Network (ANN), etc. Bayesian methods require a-priori knowledge of the data to make assumptions about the output's distribution. These methods are highly favorable since they avoid over-fitting and have a few parameters to tune. Similarly, to obtain an optimal model using SVM, one needs to select appropriate kernels based on data, whereas RF and ANN can determine the underlying structure of the data on their own. In terms of time-series prediction, RF tends to preform worse when input features are highly related. Additionally, when the data is sparsely distributed, the structure of the data is not approximated good enough by the tree partitions. Here is where different implementations of ANN, such as standard neural networks, convolutional and recurrent neural networks shine.

Furthermore, scalability drives the improvement of these learning techniques as seen in Fig. 4.1. Where traditional algorithms such as SVM or logistic regression improve in predictive performance when data is added, the performance plateaus with an increased amount of data. With the digitalization of the world, data availability increases. Training small neural networks improves the performance compared to traditional methods as more data is added.

Moreover, complex and very high accurate models are achieved with larger amount of data and with larger, deeper and hybrid neural networks. Here, *end-to-end deep learning* enables the network to internally learn new features without explicitly training a system in multiple stages. For example, consider the discrete samples of the FPL based route. Although they represent geographical locations, they have other properties as well. Some identify as airspace entry and exit-points. Others are locations of navigation aids while others classify as intermediate points where aircraft fly direct to the next waypoint, or points where convective weather is typical at a certain time, etc. Suppose all these properties are required to classify if an aircraft will deviate or not. The following pipeline structure, depicted in Fig. 4.2 could be proposed:

- Train a machine learning (ML) model to identify whether the point lies in Europe.
- Classify the type of coordinate (Entry, Exit, DCT direct-to) with another model.
- Indicate whether aircraft deviated with a third model.

However, with a large data set, end-to-end deep learning replaces the multiprocessing stages with a single neural network that can internally learn different properties of the data. The algorithm learns to generate relevant new features to learn the mapping function. If the algorithm is told which features to use, it might overlook certain important patterns. Moreover (additional) labeled data might not be available for multistage processing. Both of these advantages of end-to-end deep learning are not present in traditional ML or statistical regression techniques. Additionally, models containing complex variable interactions, such as atmospheric conditions, air traffic, airspace structure, etc. are better employed in ML algorithms. Finally, previous research have proved to be more successfully in accurately predicting time series data using ML algorithms compared to traditional statistical regression methods [35–37].
Disadvantages of neural networks however are, that they take long to train and have the most hyper-parameters to tune which influences training time as a result of parameter sweeping.

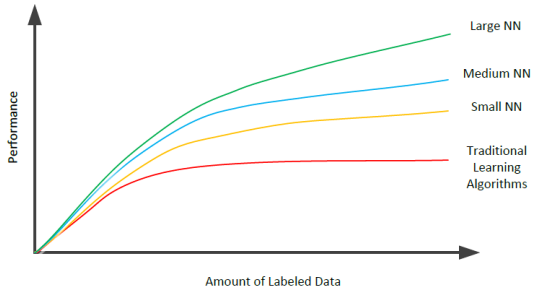


Figure 4.1: Trend of performance for different algorithms w.r.t. available amount of labeled data

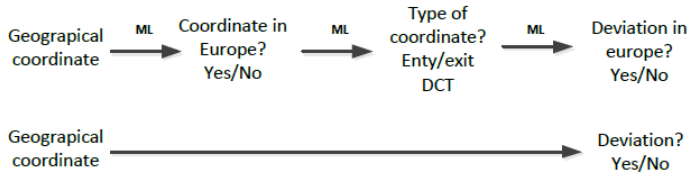


Figure 4.2: Example Multistage Processing (on top) where multiple machine learning algorithms are used to extract new features in order to predict if an aircraft has deviated or not vs end-to-end deep learning (bottom) where the model internally learns new features to classify an aircraft into deviated or not deviated

Based on the above discussion of the regression methods, ANN is chosen as the method to predict aircraft trajectories. A background on (A)NN is covered in the following sections.

4.2. Background on Neural Networks

ANNs are inspired by biological neural networks in animal brain. On the left in Fig. 4.3, it can be seen that a biological neuron consists of a cell body, or equivalently soma, and two tree branches called the axon and dendrites. At the center of the soma is the nucleus, which stores information of hereditary traits, and a plasma providing the material needed to sustain the neuron. A biological neuron receives signals (impulses) through its dendrites. The signals are generated in the nucleus of other neurons and transmitted by its axon which branches into strands. The point of contact between two neurons is at the end of these strands, called the synapse. Certain chemicals, i.e. neurotransmitters, are released at the synapse when the impulses arrive. The chemicals then diffuse across the synapse's gap and enhance or inhibit the neuron's urge to emit the impulses. The behavior of the synapse can be changed due to the passing signals and neurotransmitters. This adaptation enables the synapse to learn from its previous actions and therefore acts like a memory. In the cerebral cortex of humans, there are approximately 10^{11} million interconnected neurons, of which there are about 100 different types based on functionality. The degree of connectivity depends on the location in the brain. Furthermore, the train of short pulses enables neurons to communicate. The message is encoded on the frequency at which the pulses are emitted, which varies from 0 to 100 Hertz. Although this is relatively slow, humans can recognize a face in a matter of a few hundred milliseconds. The ability to perform such complex perceptual decision tasks is thanks to the parallel and distributed representation and computation in the brain [38].

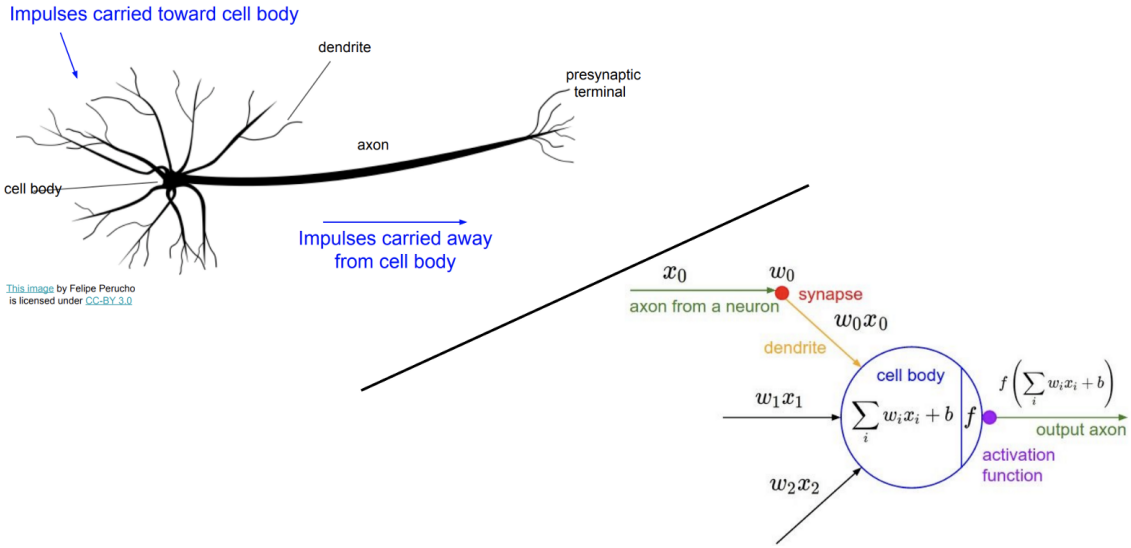


Figure 4.3: Representation of a biological neuron (left) and an artificial neuron (right) [4]

4.3. Artificial Neural Networks

In the same way as a biological neuron, the artificial computational node in Fig. 4.3 receives inputs, x_i, from different artificial neurons which are multiplied by weights w_i. The weighted sum of the signals is then calculated and a bias term b is added. The resulting value, z (Eq. 4.3), is passed through an activation function f, which controls the behaviour of the neuron activation, while subjecting z to specific bounds. The output of the activation function is simply referred to as activation, a. The parameter vector $\boldsymbol{\theta}$ is obtained by concatenating the weight matrix and the bias vector. $\boldsymbol{\theta}$ is iteratively fine-tuned in an attempt to learn the mapping function, F, between the inputs and the outputs.

$$z = \sum w_i x_i + b \tag{4.3}$$

$$a = f(z) \tag{4.4}$$

4.3.1. Activation Functions

The activation functions are analogous to the synapses and represent the rate at which the incoming signals are fired. The choice of an activation function can highly improve the performance of a NN. According to Krizhevsky et al.(2012), a network with Rectified Linear Unit (RelU) activations have proven to be six times faster than a corresponding network with tangent activation functions[39]. This is due to the gradient of tangent and sigmoid functions, which is required to train the network. The gradient becomes very small when z is either very small or very big. On the contrary, the slope of a ReLU activation is always 1 when z is positive and 0 when z is negative, permitting faster training. Table 4.1 presents the most commonly used activation functions and their derivatives in ANN, while Fig. 4.4 shows their trends.

Table 4.1: Commonly used activiation functions in ANN and their derivatives

Activation	**Equation**	**Derivative**
Step	$f(z) = \begin{cases} 0, & for\ z < 0 \\ 1, & for\ z \geq 0 \end{cases}$	$f'(z) = \begin{cases} 0, & for\ z \neq 0 \\ \infty, & for\ z = 0 \end{cases}$
Sigmiod	$f(z) = \frac{1}{1+e^{-z}}$	$f'(z) = f(z)(1-f(z))$
Tanh	$f(z) = \frac{1}{1+e^{-2z}} - 1$	$f'(z) = 1 - f(z)^2$
ReLU	$f(z) = \begin{cases} 0, & for\ z < 0 \\ z, & for\ z \geq 0 \end{cases}$	$f'(z) = \begin{cases} 0, & for\ z < 0 \\ 1, & for\ z \geq 0 \end{cases}$

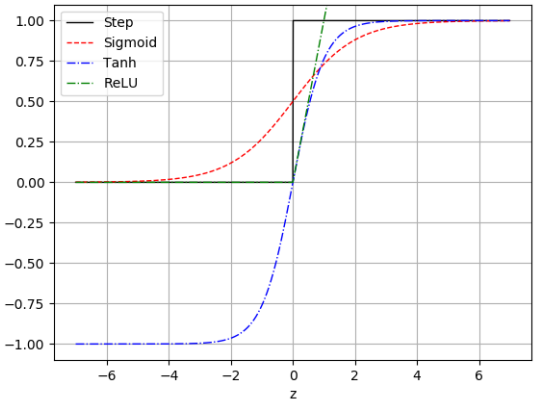


Figure 4.4: Common Activation Functions

4.3.2. Feed Forward Neural Networks

A Feedforward Neural Network (FFNN) is created by connecting multiple artificial neurons together, in which the data $\mathbf{x}$ only flows in the forward direction, i.e. from input to output $\hat{\boldsymbol{y}}$. This is also known as the forward pass. Conventionally, a standard network consist of 2 layers, the hidden layer and the output layer. However, some literature also refer to the input as an input layer. In this study, the conventional definition is maintained. Networks with 2 or more hidden layers can be considered deep NNs. Fig. 4.5 shows an example of a deep FFNN with l hidden layers, which is parameterized by $l+1$ weight matrices and $l+1$ bias vectors. Note that the activations of neurons in hidden layers are also referred to as hidden states, $\boldsymbol{h}$.

Training such a network involves two phases: forward-propagation, also known as forward pass, where activations and outputs are calculated, and back-propagation. The network learns in the back-propagation by calculating the model's Jacobian matrix and updating updating its parameters, whereby using an error-

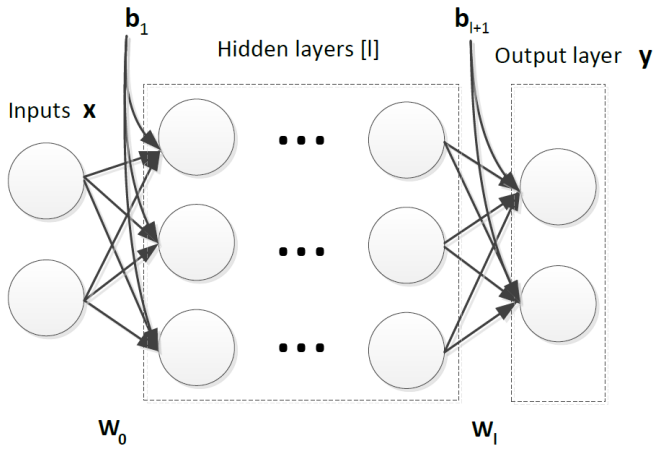


Figure 4.5: Deep feed forward neural network with l hidden layers and parameterized by $l+1$ weight matrices $(W_0,...,W_l)$ and $l+1$ biases $(b_1,...,b_{l+1})$

correction learning rule such as the squared error loss function $L(y,\hat{y})$ (Eq. 4.5). The cost function $J(\boldsymbol{\theta})$ (Eq. 4.6) is an extension of the loss function which sums over all individual training examples, $\left\{\boldsymbol{x_1^1},\boldsymbol{x_2^2},...,\boldsymbol{x_N^n}\right\}$. It provides a measure of how well the current data set is predicting the observed targets $\boldsymbol{y}$. The goal is to minimize the cost function by adjusting the parameters $\boldsymbol{\theta}$ of the network, making this an optimization problem. In order to do this, the gradients of the cost function with respect to each parameter are calculated using the chain rule from calculus. Eq. 4.7 demonstrates the gradients of the cost function w.r.t. to the weights connected between the input and the single hidden layer of a FFNN.

$$L(y,\hat{y}) = (y_i^p - \hat{y}_i^p)^2 \tag{4.5}$$

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} L(y,\hat{y}) \tag{4.6}$$

$$\frac{\delta J}{\delta w_{0j}} = \frac{\delta J}{\delta \hat{y}}\frac{\delta \hat{y}}{\delta z_2}\frac{\delta z_2}{\delta a_{1j}}\frac{\delta a_{1j}}{\delta z_{1j}}\frac{\delta z_{1j}}{\delta w_{0j}} \tag{4.7}$$

Consider the cost function to be quadratic as shown in Fig. 4.6. With the gradients calculated in the back-propagation at the current settings of the parameters, the paramter's update rule in Eq. 4.8 is used to change ω and b in the direction of the steepest descent of the cost function. α is the learning rate, which is used to control the update rate of the parameters, i.e. the step size towards the optimum. Choosing α too large or too small will determine how fast or slow the algorithm learns respectively, and introduces a risk of overshooting the direction of the steepest descent. The optimization finds the global minimum where $\boldsymbol{\theta}$ corresponds to the cost function with the lowest value. This process is called *Gradient descent*.

$$\begin{aligned} w &= w - \alpha\frac{\delta J(\theta)}{\delta w} \\ b &= b - \alpha\frac{\delta J(\theta)}{\delta b} \end{aligned} \tag{4.8}$$

However, due to non-linearities within the NN, the cost function becomes a non-convex, which does not only contain the global minimum but also local minima and saddle points. A non-convex cost function as depicted in Fig. 4.7 makes an optimization task difficult. Consequently, training the network means changing the network parameters in the direction of the steepest descent towards a *local* optimum. The value of the cost function for this local minimum, although not minimal, is very small. Unlike linear equation solvers or convex optimization algorithms used in logistic regression or SVM, a global optimum convergence is not guaranteed here. Convex optimization problems are capable of converging from any initial parameters. However, in the case of ANNs, it is crucial to initialize the weights to very small numbers and the bias terms to zeros to enable convergence. Below are the forward and back-propagation algorithms [40].

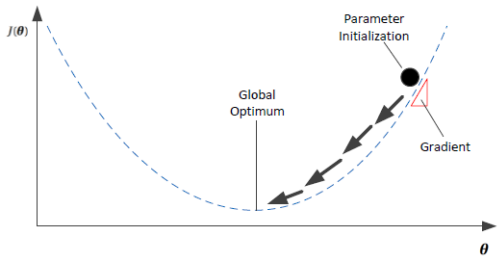


Figure 4.6: Gradient descent on quadratic convex cost function

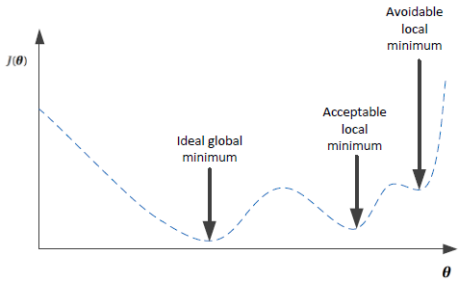


Figure 4.7: Non-convex cost function

Algorithm 1: Forward-Propagation

Require: Network depth i.e. number hidden layers, l
Require: $\mathbf{W}^{(i)}$, $i \in \{0,...,l\}$, initialized weight matrices of the model
Require: $\mathbf{b}^{(i)}$, $i \in \{1,...,l+1\}$, initialized bias vectors of the model
Require: $\mathbf{x}$, Input vector
Require: $\mathbf{y}$, Target vector

set $a_0 \leftarrow x$
for $i = 1$ to $l+1$ **do**
  $z_i \leftarrow W_{i-1} a_{i-1} + b_i$
  $a_i \leftarrow f(z_i)$
end for
$\hat{y} = a_{l+1}$
$J = L(y, \hat{y})$

Algorithm 1: Back-Propagation

$\delta a_{l+1} \leftarrow \delta L(y, \hat{y}) / \delta \hat{y}$
for $i = l+1$ to 1 **do**
  $\delta z_i \leftarrow f'(a_i) \delta a_i$
  $\delta a_{i-1} \leftarrow W_{i-1}^{\top} \delta z_i$
  $\delta b_i \leftarrow \delta z_i$
  $\delta W_{i-1} \leftarrow \delta z_i a_{i-1}^{\top}$
end for

Batch, Mini-batch and Stochastic Gradient Descent

The cost function in Eq. 4.6 sums up the error over all training examples. Gradient descent with this error-correction rule is called *batch gradient descent* (BGD). Thanks to parallel processing, the cost over the entire training set can be computed if sufficient hardware memory capacity is available. Consider a training set of $N = 5$ million training examples. Before a single step of gradient descent can be taken, all training examples need to be processed. However, faster training can be achieved by allowing gradient descent to make progress before processing the entire training set. This is done by dividing the training set into mini-batches. Suppose each *mini-batch* has a size of 1000 training examples, then the total number of batches of the training set would be 5000. Hence, the weights are updated in the direction of the steepest descent after each batch, yielding a faster training algorithm called the *mini-batch gradient descent* (MBGD). Furthermore, this means that the entire training set would be processed after the network has seen all 5000 batches, i.e. 1 *epoch*. In other words, 5000 *iterations* is required to complete 1 epoch. With a fixed mini-batch size, the training can

be continued by increasing the number of epochs. Note that epoch and batch-size are also hyper-parameters that can be tuned to improve the model's performance in terms of accuracy and speed. When the mini-batch size equals the number of training examples, N, BGD is obtained. In the other extreme case, where the mini-batch size = 1, the *stochastic gradient descent* (SGD) algorithm is obtained. Here, the network parameters are updated after processing a single training example. This moves the gradient with smaller steps towards the local minimum, whereas BGD is capable of taking larger steps with low noise. Due to the single parameter updates in SGD, it is possible for the gradient to move in the wrong direction w.r.t. the local minimum. As a result of this noisiness and update rate, SGD will take longer to train the model. Additionally, SGD will not converge but oscillate around the region of the local minimum. MBGD combines the best of both algorithms, enabling faster learning and the use of parallel processing.

4.3.3. Recurrent Neural Networks

Although FFNN works for many cases, it has its limitations. It is incapable of constructing relationships between new inputs and previous outputs of the time series due to the lack of communication through time. Moreover, it considers the input sequence to be independent and identically distributed (iid) random variables. Additionally, the rigidity of its network structure w.r.t. fixed input and output lengths poses restrictions on its implementation with different data types. This property is not desirable since sequence data, such as time series, can vary within lengths such as the reference trajectories based on the FPL. Although this issue can be solved by re-sampling the reference trajectories, this solution is undesired due to possible loss of information. Lastly, when using standard FFNN to learn a certain task, the NN will not share features learned across different positions of the input. To overcome these limitations, a recurrent neural network (RNN) can be used which specializes in processing sequential data.

RNNs have at least one feedback loop as illustrated by the folded recursive representation in Fig. 4.8. Note that this network structure has only one hidden layer. The unfolded RNN is a sequence of copies of the same neural network connected by internal states. Notice that the copies consist of a hidden layer and an output layer. This can be extended to a deep recurrent neural network (DRNN) by adding more hidden layers. From here onward, the activations of the hidden neurons are denoted as hidden states $\boldsymbol{h}_t$. In the given structure $\boldsymbol{h}_t$ does not only depend on the current input $\boldsymbol{x}_t$, but also on the previous hidden state $\boldsymbol{h}_{t-1}$. Therefore, the current hidden state acts as a memory which contains information about the entire sequence up until time step t. As a result, RNNs are able to capture long-term dependencies in a series.

Let's consider the following RNN regression example, where the output neurons have a linear activation function instead of nonlinear squashing function like a sigmoid, used for binary classification.

At current time step, t, the hidden state $\boldsymbol{h}_t$ is computed with Eq. 4.9. Here, W_{hh} and W_{hx} are respectively the weight matrices of the previous hidden activations and current inputs used to compute current hidden state. The hidden state is then multiplied by the output weight matrix W_{hy} to compute current output, $\hat{\boldsymbol{y}}_t$. The hidden state is then passed on to the next time step. The output of the current time step is given by Eq. 4.10. In both equations, $\boldsymbol{b}_h$ and $\boldsymbol{b}_y$ are bias vectors for computing the weighted sum and output respectively. Notice that the network parameters' matrices are shared across time and therefore enable back-propagation.

$$\boldsymbol{h}_t = \tanh(W_{hh}\boldsymbol{h}_{t-1} + W_{xh}\boldsymbol{x}_t + \boldsymbol{b}_h) \tag{4.9}$$

$$\hat{\boldsymbol{y}}_t = \sum W_{hy}\boldsymbol{h}_t + \boldsymbol{b}_y \tag{4.10}$$

Training the network with gradient descent is slightly modified for RNNs. In the back-propagation step, the gradients are calculated through time. This is demonstrated by an example using the regression model in Eq. 4.10 [37]. At each time step the loss is calculated. Summing up the individual losses across the entire sequence produces the objective function of the optimization problem given in Eq. 4.11. For simplicity, lets reduce the problem to a 1-dimension output and denote $J(t+1)$ as the loss at time step $t+1$. Considering $z_t = W_{hy}\boldsymbol{h}_t + \boldsymbol{b}_y$ and $\hat{\boldsymbol{y}}_{\boldsymbol{t}} = \sum z_t$ to be the output at time step t, the derivative of the cost w.r.t. to z_t is obtained in Eq. 4.12.

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2 \tag{4.11}$$

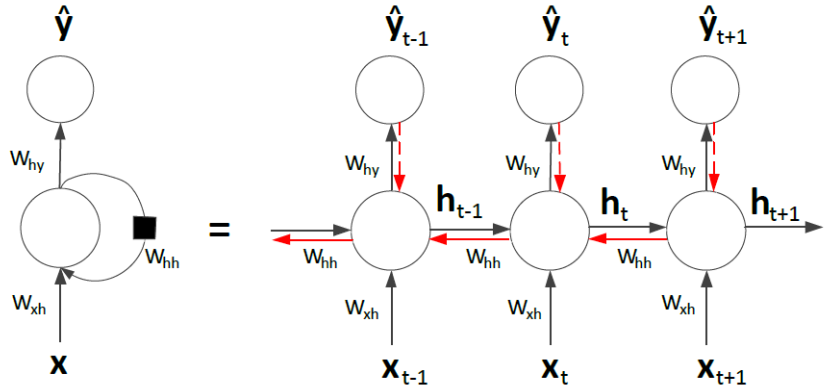


Figure 4.8: An example of a single hidden layer RNN model, with a folded RNN (left) and an unfolded model (right) displaying the sequential operations. The black arrows indicate forward-propagation while the red lines indicate back-propagation. The solid red lines are distinctive for the back-propagation through time (BPTT) algorithm

$$\frac{\delta J}{\delta \hat{y}_t} = -(y_t - \hat{y}_t) \tag{4.12}$$

Since the weight matrix, W_{hy}, is shared across the entire series, the loss can be differentiated at each time step and summed up, as seen in Eq. 4.13. The same is true for the bias b_y shown in Eq. 4.14. These two steps are represented by the dashed red lines in Fig. 4.8

$$\frac{\delta J}{\delta W_{hy}} = \sum_t \frac{\delta J}{\delta \hat{y}_t} \frac{\delta \hat{y}_t}{\delta W_{hy}} \tag{4.13}$$

$$\frac{\delta J}{\delta b_y} = \sum_t \frac{\delta J}{\delta \hat{y}_t} \frac{\hat{y}_t}{\delta b_z} \tag{4.14}$$

The next step is to compute the gradient w.r.t. W_{hh} at time step $t \rightarrow t+1$ in Eq. 4.15. It is observed that the hidden state $\boldsymbol{h}_{t+1}$ also depends on the previous hidden state $\boldsymbol{h}_t$. This means that the chain rule can be employed to estimate the slope, and since W_{hh} is shared across the entire sequence, Eq. 4.15 is rewritten as Eq. 4.16 at time step $t+1$. Here the gradient w.r.t. to $\hat{y}_{t+1}$ is calculated, whereafter back-propagation through time (BPTT) from $t \rightarrow 0$ is performed to obtain the gradient w.r.t. W_{hh}. When the gradients over the entire time sequence are summed up, Eq. 4.17 is obtained. Fig. 4.8 shows this process by the solid red arrows.

$$\frac{\delta J(t+1)}{\delta W_{hh}} = \frac{\delta J(t+1)}{\delta \hat{y}_{t+1}} \frac{\delta \hat{y}_{t+1}}{\delta \boldsymbol{h}_{t+1}} \frac{\delta \boldsymbol{h}_{t+1}}{\delta W_{hh}} \tag{4.15}$$

$$\frac{\delta J(t+1)}{\delta W_{hh}} = \sum_{k=1}^{t} \frac{\delta J(t+1)}{\delta \hat{y}_{t+1}} \frac{\delta \hat{y}_{t+1}}{\delta \boldsymbol{h}_{t+1}} \frac{\delta \boldsymbol{h}_{t+1}}{\delta \boldsymbol{h}_k} \frac{\delta \boldsymbol{h}_t}{\delta W_{hh}} \tag{4.16}$$

$$\frac{\delta J}{\delta W_{hh}} = \sum_t \sum_{k=1}^{t+1} \frac{\delta J(t+1)}{\delta \hat{y}_{t+1}} \frac{\delta \hat{y}_{t+1}}{\delta \boldsymbol{h}_{t+1}} \frac{\boldsymbol{h}_{t+1}}{\delta \boldsymbol{h}_k} \frac{\delta \boldsymbol{h}_k}{\delta W_{hh}} \tag{4.17}$$

Lastly, at time step $t+1$, the gradient of the cost function w.r.t. W_{xh} is derived with Eq. 4.18.

$$\frac{\delta J(t+1)}{\delta W_{xh}} = \frac{\delta J(t+1)}{\delta \boldsymbol{h}_{t+1}} \frac{\delta \boldsymbol{h}_{t+1}}{\delta W_{xh}} \tag{4.18}$$

Since $\boldsymbol{h}_{t+1}$ rely on $\boldsymbol{h}_t$ and $\boldsymbol{x}_{t+1}$, back-propagation needs to take $\boldsymbol{h}_t$ into account. Furthermore, taking into account all contributions from time step $t \rightarrow 0$, the gradient at time step $t+1$ is calculated with Eq. 4.19.

$$\frac{\delta J}{\delta W_{xh}} = \sum_t \sum_{k=1}^{t+1} \frac{\delta J(t+1)}{\delta \hat{y}_{t+1}} \frac{\delta \hat{y}_{t+1}}{\delta \boldsymbol{h}_{t+1}} \frac{\delta \boldsymbol{h}_{t+1}}{\delta \boldsymbol{h}_k} \frac{\delta \boldsymbol{h}_k}{\delta W_{xh}} \tag{4.19}$$

Below are the forward-propagation and BPTT of an RNN with one hidden layer [40].

Algorithm 3: Forward-Propagation RNN

Require: Network depth i.e. number hidden layers, $l = 1$
Require: $\mathbf{W}_{xh}$, initialized input weight matrix of the model
Require: $\mathbf{W}_{hh}$, initialized hidden state weight matrix of the model
Require: $\mathbf{W}_{hy}$, initialized output weight matrix of the model
Require: $\mathbf{b}_h$, initialized hidden state bias vector of the model
Require: $\mathbf{b}_y$, initialized output state bias vector of the model
Require: **x**, Input vector
Require: **y**, Target vector

for $t = 1$ to T **do**
$v_t \leftarrow W_{hh} h_{t-1} + W_{xh} x_t + b_h$
$h_t \leftarrow f(z_t)$
$z_t \leftarrow W_{hy} h_t + b_y$
$\hat{y}_t \leftarrow f(z_t)$
end for

Algorithm 4: Back-Propagation Through Time

for $t = T$ to 1 **do**
$\delta z_t \leftarrow f'(z_t)\delta L(\hat{y}_t, y_t)/\delta \hat{y}_t$
$\delta b_y \leftarrow \delta b_y + \delta z_t$
$\delta W_{hy} \leftarrow \delta W_{hy} + \delta z_t h_t^\top$
$\delta h_t \leftarrow \delta h_t + W_{hy}^\top \delta z_t$
$\delta \hat{y}_t \leftarrow f'(\hat{y}_t)\delta h_t$
$\delta W_{xh} \leftarrow \delta W_{xh} + \delta \hat{y}_t x_t^\top$
$\delta b_h \leftarrow \delta b_h + \delta \hat{y}_t$
$\delta W_{hh} \leftarrow \delta W_{hh} + \delta \hat{y}_t h_{t-1}^\top$
$\delta h_{t-1} \leftarrow W_{hh}^\top \delta \hat{y}_t$
end for
return $\delta\theta = \left[\delta W_{xh}, \delta W_{hh}, \delta W_{hy}, \delta b_h, \delta b_y, \delta h_0\right]$

Vanishing and Exploding Gradient

The nonlinear iterative nature of long-range temporal dependencies within sequences makes training RNNs a challenging task. As the number of hidden layers increases, the gradient of the loss function can either become very small or very large. Therefore, the sensitivity of the loss function to small changes increases. This phenomenon is know as the vanishing and exploding gradient. In BPTT, the factor $\frac{\delta \boldsymbol{h}_{t+1}}{\delta \boldsymbol{h}_k}$ in Eq. 4.19 indicates a sequential matrix multiplication, in which the values of the matrices are very small. For a long sequence, the gradient tends to shrink as it passes through layers. As a result, the current output is less influenced by those states early on in the sequence [37].
This weakness of the standard RNN is known as the vanishing gradient. According to Azzouni et al.(2017), RNNs can only model long-range context dependencies to $5-10$ discrete time steps between relevant input and output [35]. This limitation is solved when a Long Short Term Memory (LSTM) architecture is used [41]. Whereas a standard RNN uses a $tanh$ function to store correlations between $\boldsymbol{x}_t$, $\boldsymbol{h}_{t-1}$ and $\boldsymbol{h}_t$, LSTM networks use memory cells. LSTMs have been widely adopted in various applications ranging from natural language machine translation, speech recognition, image processing and time series predictions. The next section gives an introduction to LSTMs.

Long and Short Term Memory Algorithm

A LSTM architecture is an extension of the standard RNN. It is constructed out of memory blocks. The blocks contain a cell, $\boldsymbol{c}_t$, that encodes temporal information of previous states in the network. In addition, the memory blocks also contain gate units which are essentially neural networks with a single hidden layer. Gates control the flow of information throughout the network. Each block has an input gate, $\boldsymbol{i}_t$, that controls the flow of activations or hidden states being fed into the memory cell. The forget gate, $\boldsymbol{f}_t$, resets the memory cell

through scaling and forgets trivial correlations from previous time steps. Finally, the output gate, $\boldsymbol{o}_t$ determines the current hidden state of the network [35].

In Fig. 4.9, the previous cell state $\boldsymbol{c}_{t-1}$, previous hidden state $\boldsymbol{h}_{t-1}$ and the current input vector $\boldsymbol{x}_t$ are supplied as inputs to the memory block. The latter 2 are multiplied by their respective weight matrices W_{hf} and W_{xf}. A bias matrix is added and the sum of this is passed through a sigmoid activation function as shown in Eq. 4.20. This provides a scaling vector with values between 0 and 1, indicating least relevant to most relevant. The scaling vector is then multiplied by the previous cell state. If one of the scaling values is 0 (or close to 0), the forget gate will remove that piece of information in the corresponding component of the previous cell state. Similarly, if one of the values is 1 (or close to 1), it will keep that information. For instance, the forget gate might learn to forget the hidden states corresponding to a steady horizontal flight once the pilot initiates a descent manoeuvre. Once the information is forgotten, the cell state is updated. The input gate $\boldsymbol{i}_t$, in Eq.

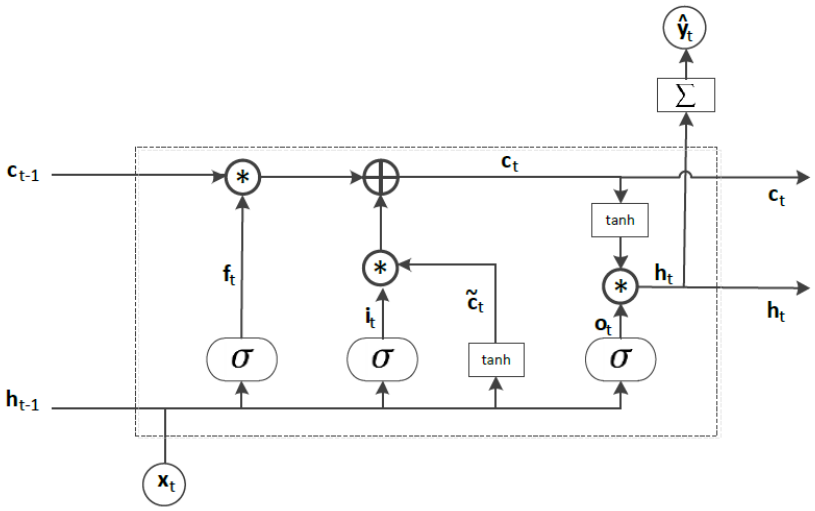


Figure 4.9: Representation of a LSTM cell

4.21, contains values between 0 and 1 and will be multiplied by the potential new cell state $\tilde{\boldsymbol{c}}_t$, calculated with Eq. 4.22, to compute the new cell state $\boldsymbol{c}_t$. The potential cell state is created by multiplying the cell state weight W_{xc} and W_{hc} to the previous input and hidden states. These are added to the previous cell state to create a new cell state in Eq. 4.23. The decision on which outputs are used, depends on an output gate in Eq. 4.24. The new cell state is passed through a tanh activation function and multiplied by the output gate, producing the current hidden state in Eq. 4.25. Finally, the output of the regression model in
Eq. 4.26 can be computed using the current hidden state.

$$\boldsymbol{f}_t = \sigma(W_{xf}\boldsymbol{x}_t + W_{hf}\boldsymbol{h}_{t-1} + b_f) \tag{4.20}$$
$$\boldsymbol{i}_t = \sigma(W_{xi}\boldsymbol{x}_t + W_{hi}\boldsymbol{h}_{t-1} + b_i) \tag{4.21}$$
$$\tilde{\boldsymbol{c}}_t = \tan(\sigma(W_{xc}\boldsymbol{x}_t + W_{hc}\boldsymbol{h}_{t-1} + b_c)) \tag{4.22}$$
$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \tag{4.23}$$
$$\boldsymbol{o}_t = \sigma(W_{xo}\boldsymbol{x}_t + W_{ho}\boldsymbol{h}_{t-1} + b_o) \tag{4.24}$$
$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) \tag{4.25}$$
$$\hat{\boldsymbol{y}}_t = \sum W_{hy}\boldsymbol{h}_t + \boldsymbol{b}_y \tag{4.26}$$

Multiple memory blocks placed in series, as illustrated in Fig. 4.10, transform the cell state into a kind of conveyor belt containing information of the whole network. When the weights of the gate units are set accordingly, correlations can be stored over long ranges in the memory cell. Similar to RNN, BPTT is used for gradient descent optimization of LSTMs. For simplicity, the following equations use $d_{variable}$ to denote the gradient of the cost w.r.t. a network parameter. The gradient from the cost w.r.t. to output weights are given by Eqs. 4.27 - 4.28. Notice that the gradients of the hidden state is computed only for the last time step T in

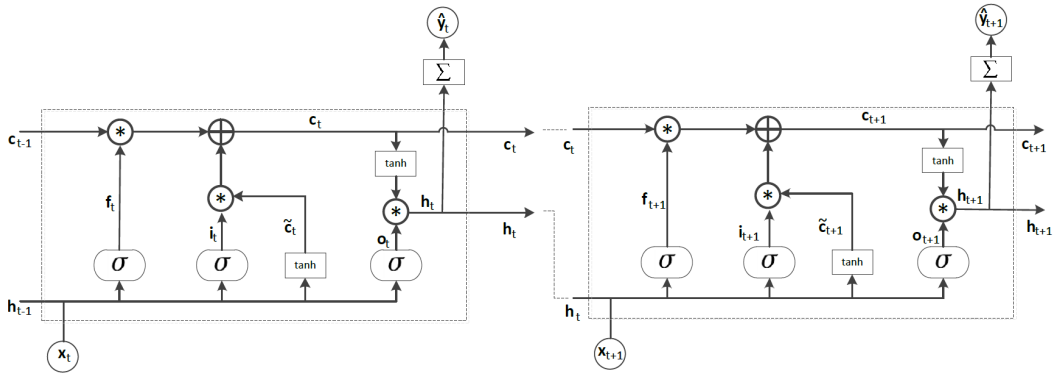


Figure 4.10: Two LSTM cells in series

Eq. 4.29.

$$d\hat{y}_t = -(y_t - \hat{y}) \tag{4.27}$$

$$dW_{hy} = \sum_t \boldsymbol{h}_t d\hat{y}_t \tag{4.28}$$

$$d\boldsymbol{h}_T = W_{hy} d\hat{y}_t \tag{4.29}$$

Equations 4.30 – 4.35 provide the derivatives of the gates at time t.

$$d\boldsymbol{o}_t = \tanh(\boldsymbol{c}_t) d\boldsymbol{h}_t \tag{4.30}$$

$$d\boldsymbol{c}_t = (1 - \tanh(\boldsymbol{c}_t)^2) \boldsymbol{o}_t d\boldsymbol{h}_t \tag{4.31}$$

$$d\boldsymbol{f}_t = \boldsymbol{c}_{t-1} d\boldsymbol{c}_t \tag{4.32}$$

$$d\boldsymbol{c}_{t-1} = \boldsymbol{f}_t \odot d\boldsymbol{c}_t \tag{4.33}$$

$$d\boldsymbol{i}_t = \tilde{\boldsymbol{c}}_t d\boldsymbol{c}_t \tag{4.34}$$

$$d\tilde{\boldsymbol{c}}_t = \boldsymbol{i}_t d\boldsymbol{c}_t \tag{4.35}$$

Furthermore, the gradient w.r.t. the shared weights across the whole sequence is obtained with equations 4.36 – 4.43.

$$dW_{xo} = \sum_t \boldsymbol{o}_t (1 - \boldsymbol{o}_t) \boldsymbol{x}_t d\boldsymbol{o}_t \tag{4.36}$$

$$dW_{xi} = \sum_t \boldsymbol{i}_t (1 - \boldsymbol{i}_t) \boldsymbol{x}_t d\boldsymbol{i}_t \tag{4.37}$$

$$dW_{xf} = \sum_t \boldsymbol{f}_t (1 - \boldsymbol{f}_t) \boldsymbol{x}_t d\boldsymbol{f}_t \tag{4.38}$$

$$dW_{xc} = \sum_t (1 - \tilde{\boldsymbol{c}}_t^2) \boldsymbol{x}_t d\tilde{\boldsymbol{c}}_t \tag{4.39}$$

$$dW_{ho} = \sum_t \boldsymbol{o}_t (1 - \boldsymbol{o}_t) \boldsymbol{h}_{t-1} d\boldsymbol{o}_t \tag{4.40}$$

$$dW_{hi} = \sum_t \boldsymbol{i}_t (1 - \boldsymbol{i}_t) \boldsymbol{h}_{t-1} d\boldsymbol{i}_t \tag{4.41}$$

$$dW_{hf} = \sum_t \boldsymbol{f}_t (1 - \boldsymbol{f}_t) \boldsymbol{h}_{t-1} d\boldsymbol{f}_t \tag{4.42}$$

$$dW_{hc} = \sum_t (1 - \tilde{\boldsymbol{c}}_t^2) \boldsymbol{h}_{t-1} d\tilde{\boldsymbol{c}}_t \tag{4.43}$$

The derivative of the hidden states at time step $t-1$ can be derived in 2 directions. In Eq. 4.44, the gradient is derived through the hidden activation and through the objective function in Eq. 4.45.

$$d\boldsymbol{h}_{t-1} = \boldsymbol{o}_t (1 - \boldsymbol{o}_t) \boldsymbol{W}_{ho} d\boldsymbol{o}_t + \boldsymbol{i}_t (1 - \boldsymbol{i}_t) \boldsymbol{W}_{hi} d\boldsymbol{i}_t + \boldsymbol{f}_t (1 - \boldsymbol{f}_t) \boldsymbol{W}_{hf} d\boldsymbol{f} + (1 - \tilde{\boldsymbol{c}}_t^2) \boldsymbol{W}_{hc} d\tilde{\boldsymbol{c}}_t \tag{4.44}$$

$$d\boldsymbol{h}_{t-1} = d\boldsymbol{h}_{t-1} + W_{hy} d\hat{y}_{t-1} \tag{4.45}$$

4.4. ANN Optimization

Once the network type is chosen and implemented, the network needs to be trained and optimized. Optimizing a NN is an iterative process, which involves: tuning hyper-parameters, selecting the right advanced gradient decent optimization algorithm, adding more data, and regularizing the learning algorithm, while evaluating the bias and variance of the model. In this chapter, the general process is presented.

4.4.1. Bias-Variance

In optimizing a NN, close attention is paid to the *generalization error*, which is the prediction capability of a learning method on an independent test-sample. Monitoring the generalization error, also called the test error, enables the selection of the right model configuration and provides a measure of the quality of model performance [5]. Eq. 4.46 provides the definition of the test error. Given the training data set τ, model F is trained with a loss function L, where the ground truth is Y and the prediction is $F(X)$. Note that X and Y are drawn from the same sample.

$$E_{rr_\tau} = E[L(Y, F(X, \mu))|\tau] \tag{4.46}$$

The test error in this equation refers to the specific, fixed training set. Therefore, the expectation of the test error is an average of test errors over random parameters settings, including randomness in training set, which produces F and can obtained by Eq. 4.47.

$$E_{rr} = E[E_{rr_\tau}] \tag{4.47}$$

Contrary to the test error is the *training error*, $\overline{err}$, representing the average loss over the training sample. The training error, given in Eq 4.6, provides a measure of how well the model is fitting the training sample. High *bias* is referred to a model that is fitting the training data poorly and high variance to a model which is over-fitting the observations. In case of high bias, a bigger network, training longer and using advanced gradient descent optimization algorithms can be opted to reduce the bias. From Fig. 4.11, it can be seen that, as the model complexity increases, the test (light red curve) -and training (light blue curve) error decrease. After an intermediate model complexity, the training error continues to decrease while the test error starts to increase. From this point onward, the model experiences an undesirable increase in *variance*. As a result, the generalization performance of the learning model degrades. Ideally, the hyper-parameters which increase the model complexity should be optimized to yield low bias and low variance.

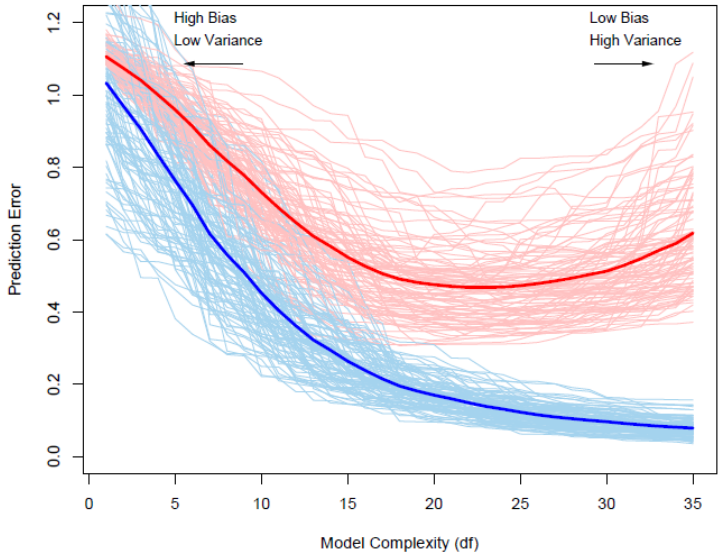


Figure 4.11: "Behavior of test sample and training sample error as the model complexity is varied. The light blue curves show the training error err, while the light red curves show the conditional test error Err_τ for 100 training sets of size 50 each, as the model complexity is increased. The solid curves show the expected test error Err and the expected training error $E[\overline{err}]$" [5]

4.4.2. Regularization

Large ANN models with low bias and high variance are capable of modeling complex function, but tend to overfit the training data. As a result, accuracy is lost on the generalization of the test set. In modern ANN design, a bias-variance trade-off is not strictly paramount. Besides adding more training data, common regularization techniques discussed in this section can improve the generalization of complex learning models.

Norm Regularization

Norm regularization is a basic method to reduce overfitting by penalizing a cost function with a norm. Popular norms used in regression analysis are L_1 and L_2. These are calculated using an equivalent of the euclidean norm. L_1 regularization tends to make the parameter matrix sparse. The type of regression is called *lasso* regression. Alternatively, *rigde* regression utilizes L_2 regularization and causes the model to spread out the magnitude of the weights.

In ANNs, the L_2 norm is generalized for matrices, called the frobenius norm. In Eq. 4.48, a penalty term is added to the cost function in Eq. 4.6, where the norm of the weight matrix is defined by Eq. 4.49. The frobenius regularization hyper-parameter, λ, is tuned for the best performance.

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} L(y,\hat{y}) + \frac{\lambda}{2N}\sum_{l=1}^{L} ||W^{[l]}||_F^2 \tag{4.48}$$

$$||W^{[l]}||_F^2 = \sum_i \sum_j (W_{ij}^{[l]})^2 \tag{4.49}$$

Performing gradient descent with regularization includes adding the regularization term to the derivatives obtained in backpropagtion. Hence the update rule is computed with Eq. 4.50.

$$W^{[l]} = W^{[l]} + \left(\frac{dJ(\boldsymbol{\theta})}{dW^{[l]}} + \frac{\lambda}{N}W^{[l]}\right) \tag{4.50}$$

Dropout regularization

The dropout technique has a similar effect as the L_2 regularization as it spreads out the weight values of the parameters. On each iteration, dropout randomly set units from a previous hidden layers to zero with some probability, p. Consequently, the nodes of the next layer spread out the values of the weights such that they do not rely heavily on specific inputs. As a results, the model employs many weights to process and predict each example and therefore reduce over-fitting. The use of dropout on weights between time steps in RNNs should be avoided in order to preserve the long-range dependencies.

4.4.3. Network Topology Optimization

In selecting an appropriate network architecture, the parameters needed to consider are:

- learning rate, α
- the number of hidden layers
- the number of hidden units (artificial neurons) in the hidden layers
- the number of epoch
- batch size
- regularization parameter, λ

These set of hyper-parameters, denoted as μ, are optimized in the iterative process of building and training an ANN.

4.4.4. Prediction Accuracy Evaluation

During optimization, the prediction accuracy of the model needs to be evaluated. Estimating the model accuracy is essential in developing an unbiased predictor in order to ensure its performance on unseen data. Large-scale data enables the partition of the data set into three parts; training (train) set, validation (val) set and test set. The training set is used to fit different model configurations, while the validation set is employed in estimating the model prediction errors and comparing them in order to select a final model. The test set is

used at the end of the analysis to assess the generalization error [5] of the model on unseen data.

k-fold cross-validation is one of the most commonly used resampling methods to estimate a model's prediction error. It is more effective in avoiding over-fitting and thus indicating whether a model is biased or not. The *nested* k-fold cross-validation procedure is demonstrated as followed. In Fig. 4.12, the data set E is partitioned into k_1 equal-sized folds. For each experiment, k_1-cross-validation uses $k_1 - 1$ folds for training and validation and one fold for testing. The $k_1 - 1$ folds data is moreover split into another k_2 roughly equal-sized parts, where $k_2 - 1$ parts are used for training the model and one fold is used for validation. Consequently, nested cross-validation enables the selection of the final model and obtains an accuracy distribution instead of a point estimate. This results in an expected accuracy of the final model. A performance measure is used

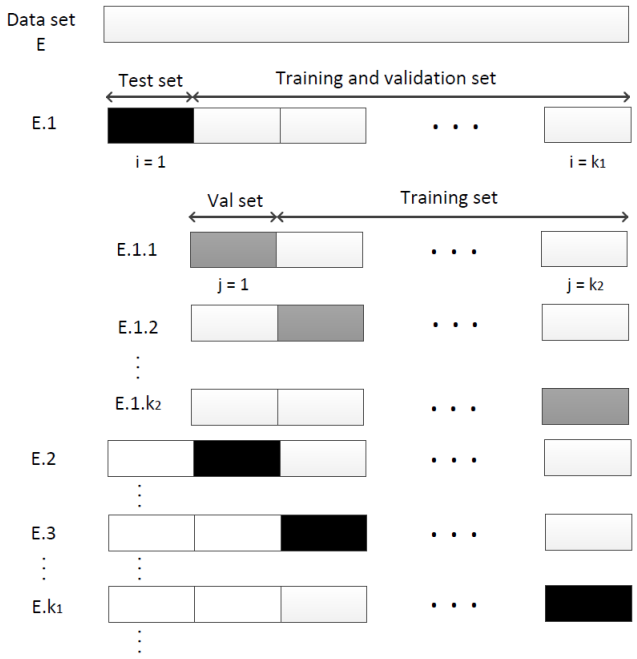


Figure 4.12: Nested k-fold cross-validation

to evaluate the effectiveness of the multi-target model during training, validation and testing. For simplicity consider y and $\hat{y}$ as a 1-dimensional predicted and actual output for $\mathbf{x}^n$, respectively. The widely used performance measures are Normalized Mean Absolute Error (NMAE) and Normalized Root Mean Squared Error (NRMSE). Both measures are scale-dependent and are computed as the sum over all individual errors of a number of training examples N. The average of the sum of errors is then taken. This is advantageous in regards to assessing the model's performance over multiple individual targets with potentially different ranges [42]. Both errors are irrespective of direction. The main difference is that NRMSE gives greater penalties to larger errors. NMAE and NRMSE can be estimated with Eq. 4.51 and Eq. 4.52 respectively.

$$NMAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{4.51}$$

$$NRMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{4.52}$$

5
Research Design

With the background information and methodology covered in previous chapters, the motivation of this research can be laid out in this chapter. The research questions are posed, followed by the objectives in this study. Furthermore, a description of the independent -and dependent variables are given. The research design is finalized by presenting the thesis' contribution and future prospects within academics and ATM respectively.

5.1. Research Questions

Based on the previous chapters, it can be concluded that there is a need for increasing airspace capacity to meet the increasing demand in air traffic. To accomplish this, one option is to improve the predictability of air traffic flow. This means that more accurate TPs are needed which will in turn reduce ATCo workload and allow ATCo to handle more aircraft. The FPL is used in all TPs. If the FPL is erroneous, it will invalidates the performance of the TP. The focus of the thesis will be on improving the predictability of air traffic flow in the horizontal plane. Consequently, the main research question is as follows;

Can the spread of ATEs and CTEs based on the FPL be reduced by predicting more accurate 2-dimensional aircraft trajectories before take-off, whereby including aircraft deviation related features in a RNN?

Considering this thesis is a data-science oriented project, the research question can be broken down into sub-questions, formulated in the following subsections:

1. Data Collection and Features Preparation
   (a) Which data set is available for predicting trajectories (**dependent variables**)?
   (b) Which features (**independent variables**) have the biggest influence on deviations and can be useful in predicting trajectories?
   (c) Which other essential features can be constructed from the available data?
2. Methodology
   (a) Which RNN architecture is most suitable for the time series characteristics of trajectories given the feature space?
   (b) Which advanced optimization algorithm can be used to improve the performance of the model?
   (c) What are the appropriate values of the hyper-parameters μ?
   (d) What are the parameters of the k-fold cross-validation?
3. Trajectory Analysis
   (a) How accurate can latitude and longitude be predicted during training?
   (b) What is the average generalization error found with k-fold cross-validation?
   (c) What is the prediction error for (deployment) test data, which are pre-processed using the same statistical properties as the training data?

4. Deviation Analysis
   (a) How does the distribution of the maximum ATE, based on the predictions of the ML model compare to that of the planned trajectories?
   (b) How does the distribution of the maximum CTE, based on the predictions of the ML model compare to that of the planned trajectories?

5.2. Research Objectives

To answer the research questions, the objectives of the research are clarified and summarized here. The main research objective of this thesis is to *predict trajectories within MUA by applying machine learning techniques and quantify and analyze their deviations w.r.t. the true trajectories.* The sub-objectives are based on the sub-questions posed in section 5.1:

1. Obtain and pre-process data that correlate most to deviations from FPL which are used as input features to the ML.
2. Develop a RNN architecture that captures the underlying structure of flown trajectories w.r.t. feature space and predicts 2-dimensional trajectory.
3. Analyze the prediction error performance of the ML model.
4. Analyze deviations from routes constructed by the ML model's output w.r.t. the actual trajectories.
5. In the post-analysis, propose improvements to ATM based on other newly gained insights.
6. Summarize the main conclusion drawn from the results and post-analysis in the report.
7. Present recommendations on how to improve the LSTM predictions.

5.3. Independent Variables

The factors found contributing to deviations are listed below.

- Atmospheric Conditions
- Convective Weather
- Change in Economic Route
- Miscommunication between an ATCo and a pilot
- Conflicting Air Traffic
- Air Traffic Conditions
- Availability of Restricted Airspace
- Airspace Structure

Based on these factors, some independent variables are constructed.

1. Atmospheric conditions
   - *wind direction*
   - *wind speed*
   - *air temperature*
   - *wind correction angle*
2. Change in economic route
   - *period of the day*
   - *estimated consumed fuel*
3. Airspace structure
   - *ICAO sector code*
   - *longitude*

- *latitude*
- *pressure altitude*
- *angle to destination*
- *distance to destination*

The only factor accounting for air traffic conditions is *airspace occupancy count*. However due to missing flight data, this feature is not used. Additionally, some other relevant features which are not based on the influencing factor are obtained. These include:

1. Aircraft parameters
    - *track angle*
    - *flight path angle*
    - *ground speed*
    - *flight path angle*
2. Miscillaneous parameters
    - *normalized flight time*
    - *wake turbulence category*
    - *distance to destination*
    - *number of trajectory samples*
    - *flight duration in MUA*

The relevance and construction of all these features are presented in Chapter 6 Note that the available data do not consist of all contributing factors and neither are all engineered features related to these factors. The relevance of some of these factors is questionable. Therefore, some data based on these contributing factors are either unavailable or unused. This is discussed in greater detail in section 6.2.4. Regardless, a deep RNN architecture might be able to capture these unavailable features or even generate its own in order to learn patterns between the inputs and outputs.

5.4. Dependent Variables

The dependent variables are the outputs whose changes are being studied. In this study, the dependent variables being evaluated are:

- Latitude prediction
- Longitude prediction
- ANN training Error

5.5. Proposal Contribution and Future Prospects

This project is part of an ongoing research project at Delft University of Technology: *Trajectory Prediction for Medium Term Conflict Detection*. The goal of this research project is to improve the accuracy of the trajectory prediction for a Medium Term Conflict Detection [43]. Data-driven post-event analysis can contribute to the ongoing research by providing high accuracy input data. Furthermore, insights of the study can be used to increase the efficient use of airspace and help improve air traffic flow management.

6
Simulation Design

In this chapter, the procedure of designing the simulation is explained. The chapter discusses the data acquisition and management tool used. Furthermore, the model input and output are presented, followed by the trajectory predictor design.

6.1. Data Acquisition and Management Tool

Data collection and pre-processing is essential in this study. Therefore, an appropriate management tool is also required. This section elaborates on the data pipelines and processing architecture used.

6.1.1. Data Acquisition

The data necessary for this study are described using Fig. 6.1, which depicts a simplified data flow in a typical ATM system. The data flow process starts with an air operator sending a FPL to NM (formerly Central Flow Management Unit). After being confirmed and accepted by NM, the FPL and a trajectory prediction based on this FPL is distributed to Air Navigation Service Providers (ANSPs) whose airspace a flight is going to traverse. Using received FPLs and weather forecasts, ANSPs strive to provide safe and expeditious flow of traffic by means of ATC and airspace regulations. Prior to a departure, a pilot receives the FPL and the Flight Management System generates a trajectory for optimal operation of an aircraft. By means of surveillance radars, ANSPs monitor the progress of the flight. The importance of Fig. 6.1 is to show the types of data required to improve the fidelity of the model used to predict trajectories. Different types of data are indicated by roman numbers. Notice that voice communications (0), if not stored digitally, are not available for use.

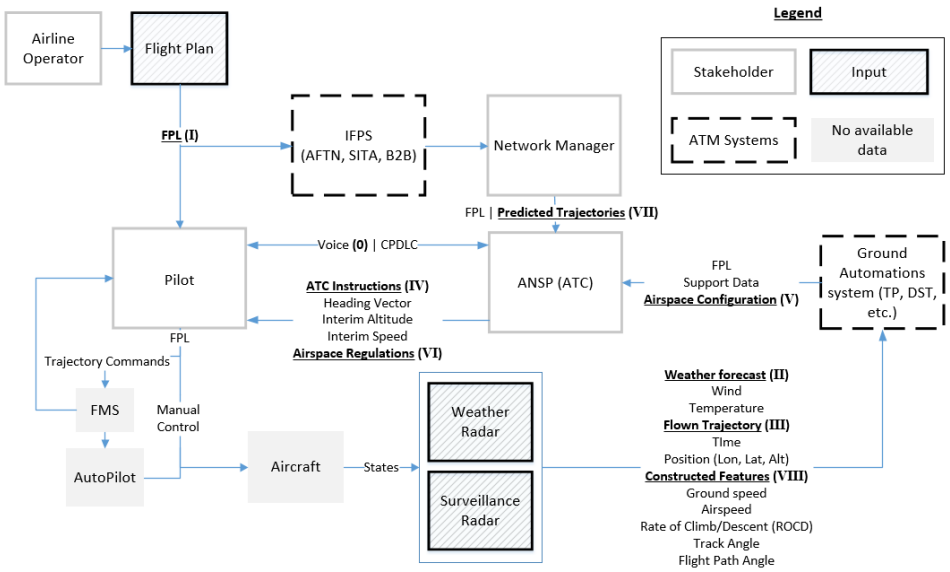


Figure 6.1: ATM communication lines

For this study, the data for 2016 was provided by MUAC, Eurocontrol, which provides services to the upper airspace users above flight level (FL) 245 over Belgium, The Netherlands, Luxembourg and North-West Germany as depicted in Fig.6.2. The obtained data, which can be traced back in Fig. 6.1, comprises flight related

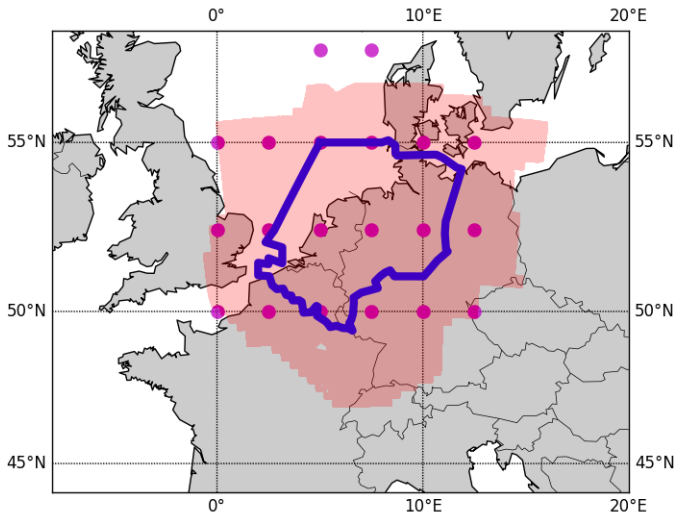


Figure 6.2: The blue line indicates the boundaries of MUAC airspace, the red area shows MUAC radar coverage, and the purple dots represent the weather forecast data points.

information (I), weather forecasts (II), flown trajectory (III), ATCo clearances (IV, see Table 6.1), airspace geometric configuration (V), airspace regulations (VI), and ALLFT+ data. Amongst other information, ALLFT+ data format includes trajectories based on the last filled flight plan (called M1 format (VII)) and flown trajectories based on surveillance data (called M3 format (III)), and can be obtained from Eurocontrol Demand Data Repository [1]. While M1 and M3 trajectories represent complete flights (from take-off to landing), trajectories flown in MUAC airspace comprise only en-route parts of these flights. Annually, approximately 2 million flights traverse MUAC airspace. In this study, only scheduled flights are considered, which comprise 88% of all flights (see Table 6.2).

Table 6.1: Types of clearances used by ATCos in MUAC

Clearance	**Abbreviation**	**Description**
Direct to	D2P	Cleared to fly directly to a specified point
Heading	HDG	Conflict avoidance clearance by issued headings
Speed	SPD	Speed clearances
Cleared Flight Level	CFL	Assigned flight level
Transfer Flight Level	TFL	Transfer to a lower/upper or external sector

Table 6.2: Number of flights in MUAC in 2016

Flight Type	**Amount**	**% of flights**
All	1,978,046	100
Scheduled	1,734,213	88

6.1.2. Data Management Tool

A data management tool is created to store and process data and create models as shown in Fig. 6.3. From the PostgreSQL database, flights passing through MUAC airspace are queried according to their date and aircraft ID. With the associated Individual Flight Plan Message Identification (IFPLID), $M1$ and $M3$ are found in text files containing ALLFT+ formats. The reference and flown trajectories are sent to the flight module where the ALLFT+ format is parsed, augmented and where new features are created. These features are generated with the 'Feature Module' and 'Meteorlogical Module'. Various mathematical methods, defined in 'external methods' assist all modules in processing the data. After processing, the data is stored in PostgreSQL database. The

pre-processed data is then sent to the 'Machine Learning Module', where one of Python's machine learning libraries, such as *Keras, Tensor Flow, Theano, Scikitlearn or Pytorch* is used for training, validation and testing. With respect to ANN, the computational load increases with data size and the number of computational operations involved. To increase computation efficiency, appropriate computer hardware (GPU) is required.

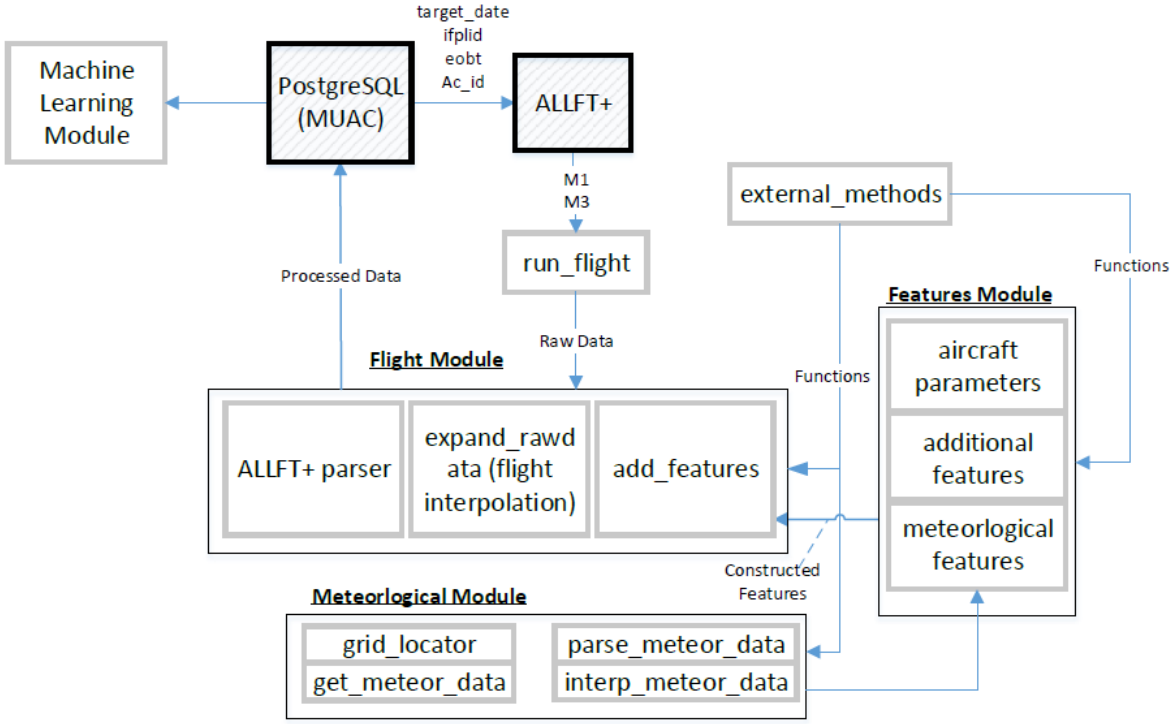


Figure 6.3: The data management tools used in study

6.2. Model Input

Inputs used in a predictive model are one of the most important factors as they directly influence output of the model. '*Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data*[1]. The section aims at generating features for predicting trajectories. Explanation and the equations used to estimate these features are therefore provided.

6.2.1. Feature Engineering

Since the objective of this study is to predict trajectories before take-off, the input features generated and constructed are based on the FPL ($M1$). This means that all features used as model input are predicted variables by ATFM systems due to the absence of real-time data before take-off. Only the construction of the targets employs the true trajectory.

Re-sampling

The $M3$ flight sequence needs to be re-sampled in order to compare the flown trajectory and the predicted route w.r.t. position and time. First, the flight segments through MUAC airspace are identified using *point in polygon* algorithm [44]. Then, the positions along the flown route are re-sampled using the average speed flown between consecutive track points. Together with the initial position, track angle and distance calculated using the average speed and FPL sample time, the direct geodetic problem can be solved[45]. Finally, two vectors of the same size are obtained for M1 and M3. This approach of sampling ensures that the speed between track points is retained.

[1] Dr. Jason Brownlee, Machine Learning Mastery

Position

The positions in M1 carry more than geographical information. Bongiorno et al. (2017) [6] identified a set of navigational point pairs that indicated over-expressed and under-expressed deviations, meaning that some navigation points classify as points where frequent deviations are observed and other points where the rate of deviations is less. Additionally, geographical locations also correlate with atmospheric conditions and nominal airspace traffic flow patterns. Therefore, it is essential to include position information as input to the model.

Ground Speed

Deviations or uncertainty in position can occur due to inaccurately estimated ground speed [31]. To enable the network to find patterns between estimated ground speed patterns and track, estimated ground speed is constructed. The average ground speed (V_g) is calculated by differentiating the distance w.r.t. time, where ds is a segment distance and dt is a segment duration (see Eq. (6.1)). The distance traveled is calculated using the *Haversine* equation [46].

$$V_g = \frac{ds}{dt} \tag{6.1}$$

Track Angle

The track angle represents the intended course of the aircraft w.r.t to the geographical north and is estimated by Eq. (6.2).

$$\chi_g = atan2\left(\frac{y}{x}\right) \tag{6.2}$$

$$y = sin(\lambda_{i+1} - \lambda)cos(\phi_{i+1}) \tag{6.3}$$

$$x = cos(\phi_i)sin(\phi_{i+1}) - sin(\phi_i)cos(\phi_{i+1})cos(\lambda_{i+1} - \lambda), \tag{6.4}$$

Here, i is the number of sample points along the trajectory, $i = 1..n-1$, λ is the latitude, and ϕ is the longitude.

Rate of Climb/Descent (ROCD)

The rate of climb/descent is the change of vertical distance meassured over a time interval.

$$ROCD = \frac{h_{i+1} - h_i}{t_{i+1} - t_i} \tag{6.5}$$

Flight Path Angle

Flight Path Angle, γ, is an angle between the aircraft's flight path and the horizontal plane and can be obtained as in Eq. (6.6). V_a is the true airspeed for which the derivation is presented later on.

$$\gamma = asin\left(\frac{ROCD}{V_a}\right) \tag{6.6}$$

Fig. 6.4 shows an example of the comparison between a FPL reference trajectory (blue line) and the actual flown trajectory (red dashed line). Lateral deviation from the FPL is clearly visible between 6 to 7 degrees longitude and 52.1 to 52.2 latitude. The same legend applies to Fig. 6.5, where the derived features are plotted. Vertical deviations can be observed in the altitude plot at the beginning and at the end of flight. The same trend can be seen in the flight path angle. Similarly, the track angle, relating to lateral deviation, shows more variation at the beginning and half way the flight. In the time domain, ground speed is indicative of longitudinal deviation.

Although the above aircraft parameters are excessively used in detecting deviations, these might not be sufficiently influential in accurately predicting trajectories. Hence additional features (see Table 6.3), based on some of the influencing factors, described in sec. 2.3, are constructed and evaluated.

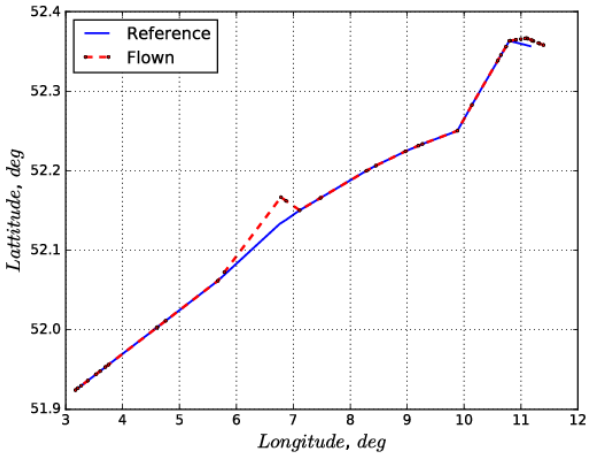


Figure 6.4: Aircraft trajectory

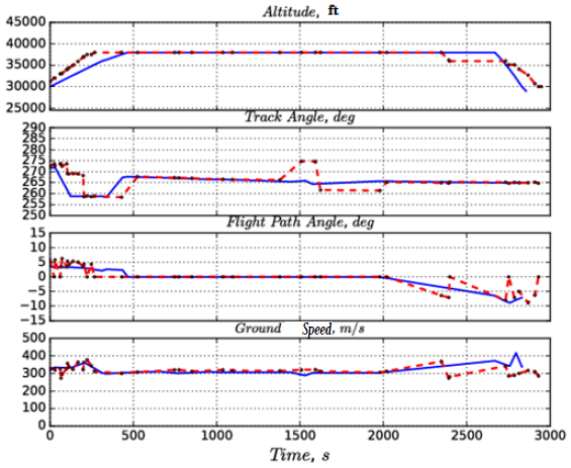


Figure 6.5: Aircraft parameters

Table 6.3: Additional features

Feature Name	Symbol
Angle to destination	X_{α}
Distance to destination	$X_{\hat{d}}$
Distance from origin	X_{OP}
Occupancy count	X_{Occ}
Atmospheric conditions	X_{w}
FPL information	X_{FPL}

Angle to destination

In the study performed by Bongiorno et al. (2017), deviation is defined as an event in which an aircraft is passing over a certain navigation point present in the planned M1 trajectory and does not go to the next as specified in the planned M1 trajectory [6]. In this study the relation between deviation and planned angle-to-destination was investigated. The planned Angle-to-destination (α) is the angle w.r.t. the destination airport measured along the current course and calculated using the spherical cosine rule in Eq. 6.7. Specifically, for each $M1^i$, characterized by the waypoints P^i_j between origin O^i and destination D^i, the segment angle-to-destination α^i_j, between 2 consecutive waypoints is illustrated on the left in Fig. 6.6. In the same figure on the right, the probability density function of α^i_j for all navigation points and points where deviation from FPL occurs is obtained in [6]. The difference in patterns show that deviations at different navigation points are not randomly distributed along the planned trajectory. According to this density function, α between $20° - 25°$ seem to be typical for deviated flights. As a result, using α, the proposed model would capture a correlation between navigation points, the α^i_j and the deviations occurring there.

$$X_{\alpha^i_j} = \arccos\left(\frac{\left(D^iP^i_j\right)\cdot\left(P^i_jP^i_{j+1}\right)}{\left\|D^iP^i_j\right\|\cdot\left\|P^i_jP^i_{j+1}\right\|}\right) \tag{6.7}$$

Normalized distance-to-destination

Bongiorno et. al. (2017) also studied the relationship between deviations and M1, by evaluating the probability density function of the normalized distance-to-destination, giving in Eq. 6.8. Here, l is the total length of $M1$ and d^i_j the distance from the navigation point to the destination D^i. From the probability density functions of d^i_j in Fig. 6.7, it was concluded that deviations occur as early as in the beginning of the flight and are not able to recover from the accumulated en-route delay. Furthermore the functions also demonstrates differences in densities between d^i_j for all navigation points and points where deviation occurred. This suggests that points where deviation occur, are not randomly distributed along the trajectory. Therefore the patterns

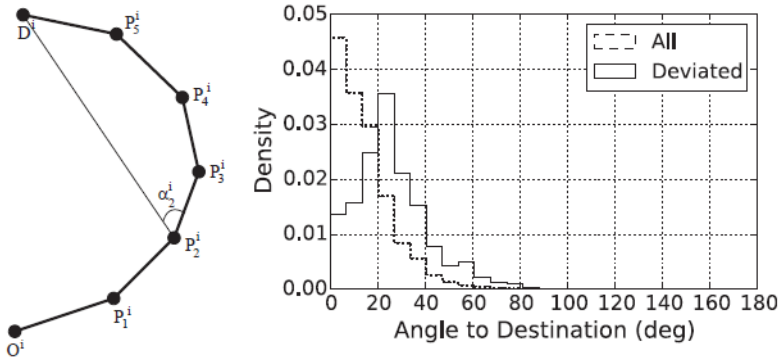


Figure 6.6: "The left panel illustrates our definition of angle-to-destination. The right panel shows the probability density function of the angle-to-destination variable estimated at a navigation point where a deviation occurs (solid line) and for all navigation points (dashed line). Data refer to the whole 334 AIRAC and the German (ED) airspace."[6]

displayed by this variable will be learned by the model.

$$X_{\hat{d}_j^i} = \frac{d_j^i}{l^i} \tag{6.8}$$

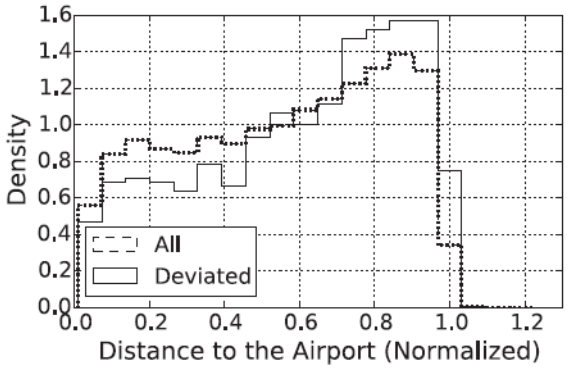


Figure 6.7: "Probability density function of the normalized distance from the arrival airport of the navigation point where a deviation occurs (solid line). As a comparison, the dashed line is the probability density function of the normalized distance from the arrival airport of all navigation points crossed by the flight. Normalization is obtained by dividing the distance along the trajectory by the flight length, thus 0 corresponds to the arrival airport and 1 to the departing airport. Data refer to the whole 334 AIRAC and the German (ED) airspace."[6]

Distance from origin

It is observed from statistics that intercontinental flights experience greater bias in TPEs when deviation is induced early on in flight. The underlying structure of the bias is an important characteristic which needs to be captured by the model, since the enroute portion of the deviated trajectory is being predicted. In order to make a numerical distinction between intercontinental and intracontinental flights, the planned distance from the origin ($X_{OP_j^i}$) is obtained. The distance between each sample P_j^i of $M1^i$ and the departure airport O^i is obtained using the *Haversine* equation [46]. It is expected that there is a correlation between X_{OP^i} and deviations in the vertical and time dimension.

Atmospheric Conditions

Atmospheric conditions can impact the flight intent of an aircraft. Wind vectors with significant magnitudes can also lead to lateral and vertical deviations during climb/descent. Furthermore, tail and head winds affect the ground speed of the aircraft, leading to longitudinal deviations. For these reasons, forecasted wind information is added as input to the model. Although the actual atmospheric conditions might be different in real-time, a correlation between deviations and forecasted conditions, reflecting actual conditions, might be

captured. Wind speed, wind direction and air temperature are received by MUAC at the geographic locations depicted in Fig. 6.2 and at the following FLs; 50, 100, 180, 240, 300, 320, 340, 360, 390, 530. The atmospheric conditions are estimated at a six hours interval. Only conditions above FL 245 will be considered in this analysis.

For each $M1$ sample, forecasted atmospheric conditions should be known at that location. Considering the grid created by the measurement points, bilinear interpolation is employed to re-sample the local atmospheric forecasts. It uses the distance-weighted average of 4 measurement points enclosing the sample in 2D space. In the case of an altitude profile, tri-linear interpolation is used [47].

True Airspeed

In aircraft separation standards, longitudinal separation between aircraft on the same track may be maintained with speed control and the mach number technique. Furthermore, separation may be established by requiring aircraft to arrive over a geographical location at a certain time, which is a time-based longitudinal control. If a change in speed control or time is issued by ATC, the true trajectory will deviate from the planned trajectory. Therefore it is essential to utilize speed information, as it relates to deviations.
With knowledge of wind conditions, the true airspeed (V_a) can be calculated by subtracting local wind vector (V_w) from the ground speed vector (V_g) from as seen in Eq. 6.9. These vectors can be decomposed into North (V_N) and East (V_E) components with Eq.6.10 in the local reference frame of the aircraft, which is depicted in Fig. 6.8. By subtracting the corresponding north and east components of wind from the ground speed and taking the square root of these results, the true airspeed is obtained. With the true airspeed, the true heading ψ can be estimated with Eq. 6.11. Finally, the wind correction angle, Λ, can be calculated by subtracting ψ from track angle, χ_g, in Eq. 6.11.

$$\mathbf{V}_a = \mathbf{V}_g - \mathbf{V}_w \tag{6.9}$$

$$\begin{aligned} V_N &= \mathbf{V} cos\chi \\ V_E &= \mathbf{V} sin\chi \end{aligned} \tag{6.10}$$

$$\begin{aligned} \psi &= asin(\frac{V_{a_E}}{V_a}) \\ \Lambda &= \chi_g - \psi \end{aligned} \tag{6.11}$$

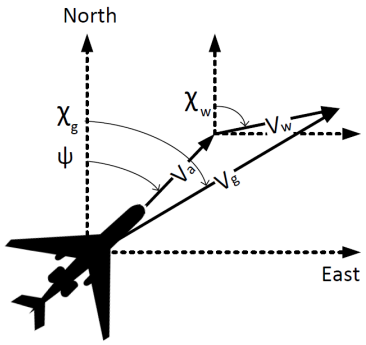


Figure 6.8: Aircraft velocity vector and wind velocity vector projected in the local reference frame

Capacity

Capacity of a sector depends on the workload experienced by an ATCo in that sector. According to Szamel et al.(2015) , ATCo workload is not only influenced by the number of aircraft handled by ATCo, but also by air traffic complexity or complexity factors[48]. Aircraft can be rerouted when traffic demands are too high. Rerouting means deviating from the FPL and is therefore seen as a reason for deviation. To evaluate deviations caused by capacity, complexity factors influencing airspace capacity should be considered. For this study, predicted occupancy count (Occ) is obtained from the data.

FPL

From the FPL, ordinal data (containing a natural sequence) and categorical data are collected as potential inputs to the proposed model.

Period of day

Bongiorno et al. (2017) [6] found that aircraft are more likely to deviate in the German airspace during the night than during day time [6]. The anti-correlation of -0.83 found in his study suggests that in low traffic conditions, the fraction of deviated aircraft is larger comparing to high traffic conditions. Bongiorno et al. (2017) attributes this observation to the possibility of change in economic route, where directs are issued. Hence, period of the day is added to the model feature space. This ordinal variable classifies the planned flight in one of the 6 hour intervals of the day.

Aircraft Type

Different aircraft types operate in different ways. Each has its own performance characteristics. Various aircraft performance models have aircraft type as input to their models, enabling them to find statistical patterns. It is assumed that the nature and extent of deviations from the FPL will therefore be correlated with the performance characteristics of an aircraft. For this reason the aircraft type can be used as a feature.

Wake Turbulence Category

Aircraft sizes determine the speed clearances given to the pilot by ATCo. Although estimated speed and aircraft type are already considered as potential features, the wake turbulence category (wtc) which is based on the mass of an aircraft can be taken into consideration. Wake turbulence vertices are invisible counter-circulating masses of air generated at the wing tips of jet aircraft. These vertices decay slowly and their strength depends on the speed and size of the aircraft, incl. weight and wingspan. Aircraft encountering wake turbulence are at risk of experiencing induced roll, loss of height or rate of climb. Time based separation standards are therefore imposed on departing and arriving aircraft by ICAO to ensure safe flight. The minimum separation standards for IFR flights are superpositioned by various separations requirements and therefore are correlated to wake turbulence minima. The minima are based on the grouping of aircraft types into three categories according to the maximum certificated take-off mass [11] which are itemized below. Therefore an expected underlying relation is seen between aircraft's wtc and ATE.

- Heavy – aircraft types of 136,000 kg or more
- Medium – aircraft types less than 136,000 kg but more than 7,000 kg
- Light – aircraft types of 7,000 kg or less.

Aircraft Operator

Each aircraft operator has particular strategies in determining the most efficient operations for their flight in an attempt to reduce travel time and fuel consumption. Even though ATCo's first obligation is to ensure seamless and safe flight, *direct-to*, for instance, can be issued in order to expedite traffic flow and therefore helping aircraft operators to save expenses.

Aggregation of Features

In Table 6.4, all potential features are summarized by presenting their source, type of feature, symbol and abbreviation and unit. Some of the features are normalized with non-linear transformations as a result of the geometric relationship with the output variables. These sine and cosine basis functions also evade the discontinuities at $\pm\pi$. The resulting transformation improves the correlation with the model's output.

6.2.2. Feature Selection

Feature selection aims at reducing the input space of a model. In this section the importance of having a low dimensional feature space is explained. Then, a visual approach and an unsupervised machine learning technique is used to select the relevant input features out of the potential features discussed in the previous section.

Table 6.4: Constructed features

Source	Feature	Abbreviation-Symbol	Unit
Allft+: M1	Normalized time	–	–
	Airspace sector	$sector$	–
	Longitude	lon	deg
	Latitude	lat	deg
	Pressure altitude	PA	ft
	ground speed	V_g	$\frac{m}{s}$
	Track angle	$TA, cos(\chi_g), sin(\chi_g)$	rad
	Flight path angle	$FPA, cos(\gamma), sin(\gamma)$	rad
	Occupancy count	Occ	-
	Angle-to-destination	$X_\alpha, cos(\alpha), sin(\alpha)$	rad
	Distance to destination	$X_{\hat{d}}$	–
	Distance from origin	X_{OP}	m
Weather Data	Wind direction	$wd, cos(\chi_w), sin(\chi_w)$	rad
	Wind speed	ws, V_w	$\frac{m}{s}$
	Wind correction angle	$wca, cos(\Lambda), sin(\Lambda)$	
	Air temperature,	$temp\text{-}T$	K
Filed Flight Plan	Period of the day	pod	–
	Weekday	dow	–
	Aircraft operator	aco	–
	Aircraft type	$actype$	–
	Wind turbulence category	wtc	–
	Number FPL sample	$n_{samples}$	–
	Flight duration in MUAC	t_{muac}	min
	Total consumed fuel	f_{cons}	l

High Dimensional Feature Space

Predictive models are designed based on learning from a dataset. They are only valid in the range of the available training set. Irrespective of the model, generalization on unseen data during the testing phase is a difficult task. Some traditional algorithms use interpolation for this purpose. They assume that the output of the new input should be approximately the same as the output of the nearest training point.

Intuitively, one would say that having more dependent variables that correlate well to the model's outputs would increase the performance of the learning algorithm. This is however not the case. Next to being counter-intuitive in geometrical space, high-dimensional spaces deteriorate the performance of models. Distance norms, Gaussian kernels and other techniques become unsuitable when increasing feature dimension. For nonlinear models, increased dimension results in a lack of model identifiability, instability, over-fitting and numerical instabilities [49]. This is known as the curse of dimensionality, meaning that the complexity of the problem increases exponentially with its dimensions. More training observations will be necessary to cover a high dimensional input space. Illustrated in Fig. 6.9 below, it is easy to see that if 10 observed data points are needed to cover 1-dimensional space, as in the 1-dimensional grid, 100 observations are required for 2-dimensional space and 1000 observations for 3-dimensional space in order to generalize well [7]. Hence, the dimension of the input space needs to be reduced to avoid the curse of dimensionality, to increase computational efficiency and to reduce errors [50]. This can be done by removing redundant elements or correlated variables from the feature data set.

Visual Approach

Although the total number of available features is small, some might have a high collinear relationship with others. In order to evaluate this, a visual approach is firstly explored. A correlation matrix is employed, representing all flights flown through MUAC airspace in February 2016. Note that statistically insignificant observations have been previously filtered out. For instance, on average, the flight duration in MUAC airspace is approximately 20 minutes with an average of 10 samples per flight. Flights with fewer than 3 samples tend to have insignificant flight time and therefore have been filtered from the data set.

Figure 6.9: The increase of configuration as the number of dimensions increase [7]

Due to the non-linear data, Spearman's rank correlation method [51] is used to initially observe which features are more or less important. In Fig. 6.10 the correlation matrix of the continuous variables are shown. It can be seen that some features are visibly highly correlated and others not. For instance temperature has a strong negative correlation with pressure altitude. Although insightful, the heat map is quite complex and therefore not the best approach in selecting features.

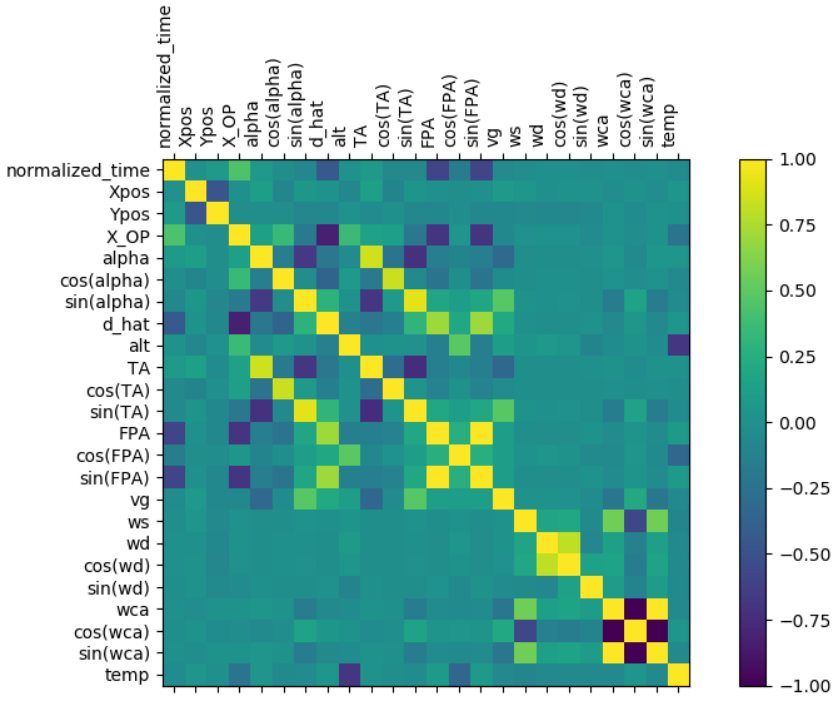


Figure 6.10: Correlation matrix of continuous input -and output variables of all scheduled flights in MUAC in February 2016

Principle Component Analysis

Principle Component Analysis (PCA), unlike the correlation matrix, enables feature selection and dimensionality reduction while retaining most of information of the data. The unsupervised machine learning algorithm seeks to transform variables, measured in their assumed orthonormal basis, to values associated with principle components of the original data set. As a result, highly correlated irrelevant data are filtered from the data set. This is achieved by changing the naive basis into another linear combination. The unit direc-

tions of the new basis correspond to the largest variance, making this a variance maximization problem. One of the methods used for PCA is called *eigenvector decomposition*. In eigenvector decomposition, the linear transformation applied in Eq. 6.12, where P is a rotational orthonormal matrix containing the principle components of X. Eq. 6.12 is only true when the covariance matrix C_Y is a diagonal matrix. From linear algebra it is known that any symmetric matrix is diagonalized by an orthonormal matrix of its eigenvectors. Therefore P is selected as a matrix where each row corresponds with the eigenvectors of the covariance matrix C_X [52]. With Spearman's rank correlation matrix R, the covariance matrix is calculated using equation 6.13, where σ_X is a diagonal matrix with standard deviations on the main diagonal and zeros everywhere else.

$$Y = PX \tag{6.12}$$

$$C_X = \sigma_X R \sigma_X \tag{6.13}$$

The dimensions are reduced by observing the cumulative explained variance q which corresponds to the cumulative highest to lowest eigenvalues of covariance matrix X. The dimensions at $q \geq 0.95$ is the number of relevant principle components to be used in the model. Fig. 6.11 shows that the number of principle components of the continuous variables is 14.

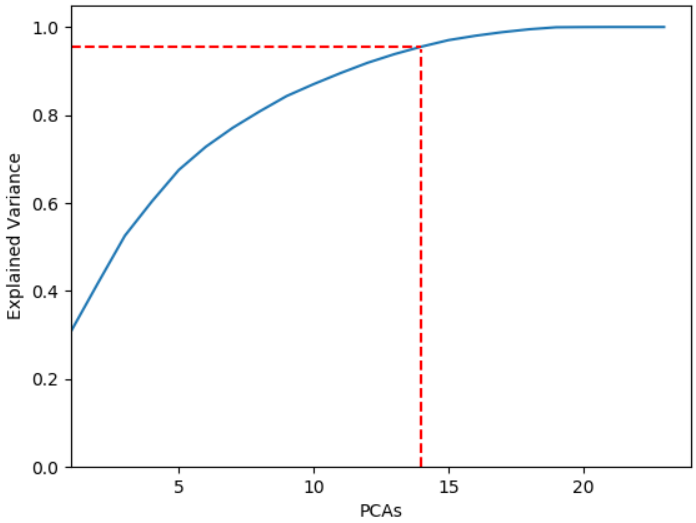


Figure 6.11: Principle components vs explained variance

6.2.3. Input Transformation

Before PCA can be performed, the continuous variables $X = [x_0, x_1, ..., x_n]$ need to be rescaled due to different scales in which the measurements were taken. Furthermore categorical and ordinal data need to be one-hot and integer encoded.

Scaling

There are different types of rescaling techniques; *Normalization* and *Standardization*. In *min-max* normalization, Eq. 6.14 is used to scale features X to $X^* = [x_0^*, x_1^*, ..., x_n^*]$ which fall in a shared range $[0, 1]$ [50]:

$$x_i^* = \frac{x_i - min(x_i)}{max(x_i) - min(x_i)} \tag{6.14}$$

Applying standardization using Eq. 6.15, rescales X to X^* with mean μ and standard deviation σ. For this study, standardization is employed.

$$x_i^* = \frac{x_i - \mu}{\sigma} \tag{6.15}$$

Encoding

Integer encoding can only be used for ordinal variables, such as period of day (pod), day of the week (dow) and wake turbulence category (wtc), because of the natural ordering they possess. After integer encoding, standardization in Eq. 6.15 is used to rescale this data.

The most interesting categorical features, after analyzing their box-plot distribution is the airspace sector. *One-hot-encoding* is employed to convert this category into binaries.

Using integer encoding for categorical variables suggest that the model will perceive a natural order within the data. This leads to poor performance and unexpected results. One-hot-encoding puts the categories around an equally spaced circle. As a result, the distance observed between each pair and the center is equal. Last but not least, some flights are missing the variable of aircraft operators and are represented as NaNs. These are also one-hot encoded.

6.2.4. Discussion Feature Engineering

Restated below are the 8 main factors that can contribute to aircraft deviating from its FPL are listed below.

- Atmospheric Conditions
- Convective Weather
- Change of Economic Route
- Miscommunication between an ATCo and a Pilot
- Conflicting Air Traffic
- Complex Air Traffic Conditions
- Availability of Restricted Airspace
- Airspace Structure

From these 8 influential factors; convective weather, ATCo and pilot miscommunication, conflicting Air Traffic and availability restricted airspace related data are not available and therefore cannot be used in estimating the model parameters. However, this is not a setback due to the objective of the study, which is to predict trajectories pre-departure. Most of these factors might be more useful in real-time applications, for example where the objective is to forecast the trajectory of a flight based on prior known trajectory data and a certain look ahead time. Additionally, it is questionable if miscommunication between ATCo and pilot could be used as an input feature. In a general sense, communication related features could be constructed where the model could learn miscommunication patterns between ATCo and pilot. However, the frequency of miscommunication is so small that it would require a highly complex model to distinct the pattern and learn from it. In the case of RNN, this might result in a very huge network and an epoch increased by multiple factors. This is not desired in training a neural network. Moreover, traffic complexity based on airspace geometry (incl. distribution of aircraft in airspace and their speed and direction relative towards each other) described by [48] should be considered for trajectory forecasting or different machine learning tasks. In terms of predicting a trajectory at $t = 0$, knowledge of these data will not be available and therefore could not be employed in the deployment of the trained model. One might argue for the use of predicted complex air traffic conditions related features. However, since there is still evidence of large deviations, the use of planned relative aircraft data to predict the trajectories might reduce the fidelity of the model.

Furthermore, there is an uncertainty in terms of the tactical occupancy count. It appears that the Allft+ database does not contain all flights flown within MUA. On an average, 12% of flights each day are missing due to missing data in MUAC Postgres database and simply because $M1$ of certain flights could not be found in $Allft+$ database. Therefore it is unclear as to use the tactical capacity as input. Additionally, MUA sector ICAO code also relates to the airspace structure and therefore can be added to the input space. The *point in polygon* approach however only looks at elementary airspace sectors. Some elementary sectors might have been combined which dynamically changes the airspace structure. The trained model will be invalid during deployment in case of newly combined elementary airspace. This is also a factor to take into consideration when selecting the final features.

To conclude, instead of using planned traffic conditions, which could reduce the model fidelity, the model uses input features in which other statistical characteristics can be found. The patterns within these conditions will be captured by the learning model in predicting aircraft trajectory.

6.3. Model Output

Based on the TPE statistics found in section 2.5, all off-track deviations are filtered out of the data set, since they relate to unpredictable changes to aircraft intent. Furthermore, the outputs (ϕ, γ) of the model need to be transformed based on their statistical behaviour. Fig. 6.12, shows the trend of longitudinal profile of random flights in MUA. As the figure shows, the slope between successive points is non-linear. Furthermore, an apparent change in level is present, primarily due to airspace entry points. This behaviour is also observed in the latitude plot in Fig. 6.13. In cases of single long time series forecasting, the difference in successive points would be selected as output variable in an attempt to stabilize the mean. However in this study, it is chosen to also add the time series differences as output variables, primarily because the inverse transformation would not be possible to obtain the actual entry coordinates of the flight. Despite this fact, predicting the sequence successive difference, together with the trajectory coordinates will enable the model explore the changing level patterns and also the smooth trajectory profiles. Hence, the complete output representation of the model is $(\phi, \gamma, \delta\phi, \delta\gamma)$.

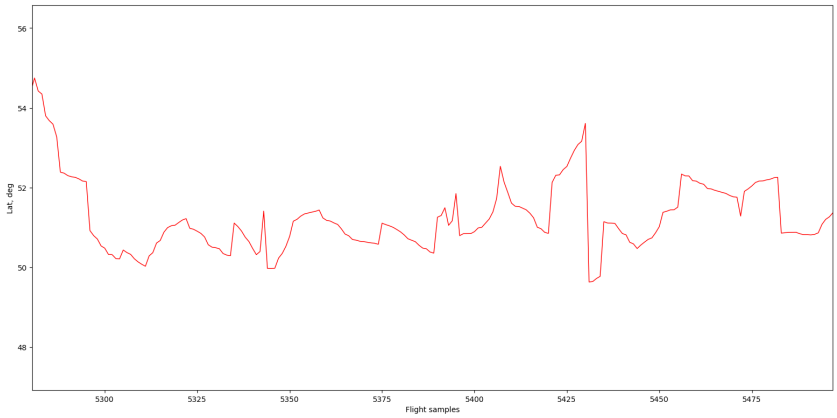


Figure 6.12: True longitude, where the jumps represent different flights in MUA 2016

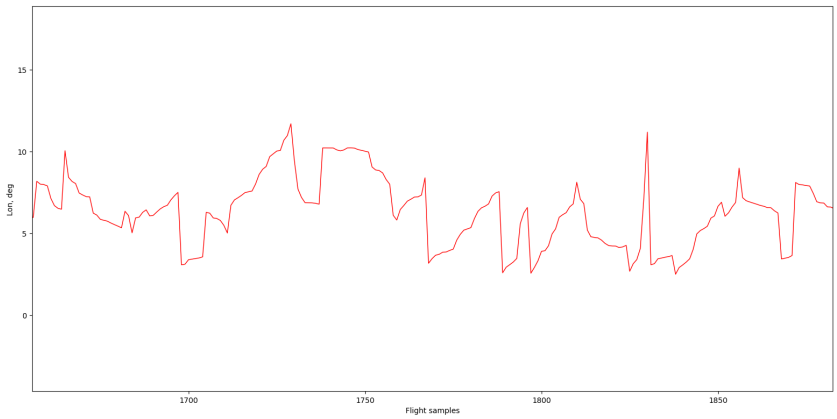


Figure 6.13: True latitude, where the jumps represent different flights in MUA 2016

Finally, all outputs are scaled between 0 and 1. Therefore the model is capable of predicting the dependent variables within the same range. Note that values are rescaled after predictions. The sequences are then padded with zero to equal length. Because of that, parallel computing and mini-batches can be used during training.

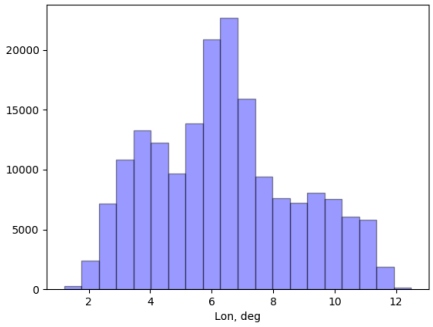


Figure 6.14: Longitude histogram of random flights in MUA 2016

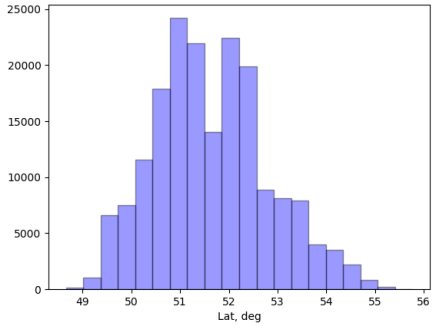


Figure 6.15: Latitude histogram of random flights in MUA 2016

6.4. Trajectory Predictor Design

In this section the process of designing a LSTM trajectory predictor is outlined. It starts with the simulation setup followed by its initialization and the optimization algorithm.

6.4.1. Simulation Setup

The simulation setup is presented by Fig. 6.16, where the flight observations are passed through a PCA module, then padded to equal length and subsequently fed to a deep LSTM model. In each of the LSTM layers a $tanh$ activation is used. After the LSTM layers, the data is fed through dense layers with RelU activation function, which predicts the true longitude, latitude, pressure altitude and the time series differences of the flights. The loss function used here is mean squared error.

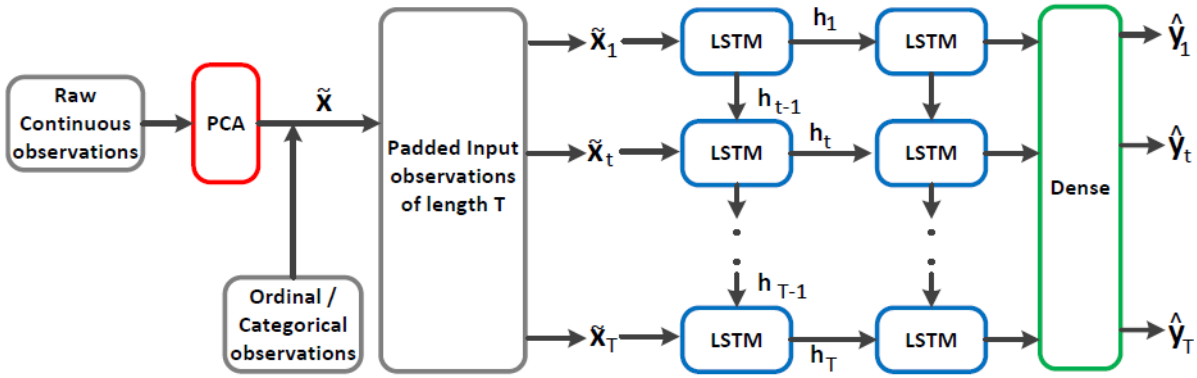


Figure 6.16: Simulation setup

6.4.2. Initialization

Training the model is an iterative process where the network parameters need to be specified to some initial point. Selecting a good initial point is crucial, since the initialization can determine if the algorithm converges or fails. In the former case, the initial point can also determine how fast the model converges. Generally, biases are initialized to zeros and the weights are drawn randomly from a Gaussian or uniform distribution. Initializing to random values ensures that symmetry is broken, meaning that two hidden units with the same activation function and connected to the same input do not compute the same function. This ensures that the input and gradient patterns are not loss in the null space of forward propagation and of back-propagation respectively [7].

The weights of the baseline architecture are initialized according to Xavier initialization also know as Glorot initialization. The scheme ensures that the weights are not too small or too large as the signal passes through many layers. In the fully connected architecture, the weights are drawn from a normal distribution with zero mean and variance as shown in equation 6.16. Here, m is the number of inputs and n the number of outputs a network layer.

$$W_{i,j} \sim N\left(0, \sqrt{\frac{2}{m+n}}\right) \tag{6.16}$$

6.4.3. Adaptive Momentum Estimation

The selected optimization algorithm is the adaptive momentum estimator or simply *ADAM*. For each parameter in the model, *ADAM* computes adaptive learning rates α_{ij} by storing and utilizing both exponentially decaying average of past squared and current gradients (v_t and m_t), shown in Eqs. 6.18 and 6.17 respectively. This ensure a smoother gradient path and a momentum in the steepest descent.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{6.17}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{6.18}$$

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{6.19}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{6.20}$$

The first and second moments of the gradients (v_t and m_t) are then bias-corrected, using Eqs. 6.20 and 6.19, due to their inherent 0 biased nature after 0 initialization. The update rule from Eq. 4.8 is then transformed to Eq. 6.21.

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{6.21}$$

7
Simulation Results

In this chapter, the results of the optimization are presented, followed by an analysis of the model predictions. Lastly the robustness of the model is evaluated using unseen data as deployment data, where the deviations of their predictions w.r.t. to the flown trajectories are calculated and presented.

7.1. Optimization Results

Optimization involved tuning the hyper-parameters in search for the combination of hyper-parameters that is most consistent and achieves the best model performance.

From statistical insights, it appears that the set of hyper-parameters to optimize pressure altitude is considerably different. This might be due to the nature of descending and ascending profile of the pressure altitude as opposed to the quasi-constant and or monotonic trends experienced in latitude and longitude. The loss function combination optimization will take a different and longer path to convergence in terms of the longitudinal and lateral variables. In light of this, it is chosen to optimize the model on the latter two, changing the focus to accurately predicting lateral and longitudinal trajectories of aircraft. From these predictions the cross-track and along-track errors can be estimated. The final model consists of 4 stacked LSTM layers with each containing 125 hidden units. The activation functions used in these hidden layers are hyperbolic tangent functions. The training data was passed through the model 30 times (epochs), using a batch size of 32 flights and whilst updating the weights with a learning rate of 0.0001.

Table 7.1: Hyper-parameters values final model

Hyper-parameter	**Values**
Optimizer	ADAM
Epochs	{30}
Number of hidden layers	{4}
Units per hidden layer	{125}
activation function hidden layers	$\{tanh\}$
activation function output layers	$\{relu\}$
Batch size	{32}
Learning rate	{0.0001}

In order to validate and to ensure an unbiased estimator, a 10−fold cross-validation is explored, using a total number of 1,147,746 filtered and pre-processed flights. The root mean squared error and mean absolute error performance of each fold is presented in table 7.2 and table 7.3 respectively. Fold 2 and 10 have slightly higher error in longitude and latitude respectively compared to all other folds. This is due to the use of randomness in the stochastic optimization algorithm, where the network is initialized with small random weights and at each epoch the batches are shuffled to explore different parts of the loss function. Therefore the search process from initial state to final state might end up in a less desirable search space when training is terminated. This is also visualized by looking at the loss functions of longitude variable recorded during the validation of fold 2, plotted in Fig. 7.1. It can be seen that the validation loss has a high oscillating amplitude. This can be caused by a learning rate that is too high or the knowledge of being stuck in a local optimum. However, since the training loss and the validation loss have a decreasing trend towards the end of training, it can be assumed that the learning rate is too high and that there are more patterns to be discovered. Therefore reducing the learning rate and training the model longer will improve the model's performance. This is

even more crucial when using smaller batch sizes, since stochastic gradient descent and smaller batch sizes will have a hard time converging. However, reducing the learning rate means slower convergence and longer training time. Alternatively, increasing the batch size and training longer while maintaining the same learning rate will also have similar effect according to Smith et al. [53]. This is true for all folds. The average of the performance metrics used in the 10-fold cross-validation are presented in table 7.4.

Table 7.2: RMSE 10-fold cross-validation

variable	**fold 1**	**fold 2**	**fold 3**	**fold 4**	**fold 5**	**fold 6**	**fold 7**	**fold 8**	**fold 9**	**fold 10**
ϕ lat, deg	0.153188	0.156799	0.163234	0.150923	0.147694	0.152967	0.158649	0.152461	0.151647	0.156564
γ lon, deg	0.253114	0.266973	0.254756	0.252916	0.253695	0.253788	0.252925	0.252628	0.253642	0.254510

Table 7.3: MAE 10-fold cross-validation

variable	**fold 1**	**fold 2**	**fold 3**	**fold 4**	**fold 5**	**fold 6**	**fold 7**	**fold 8**	**fold 9**	**fold 10**
ϕ lat, deg	0.114374	0.118830	0.124834	0.112979	0.109433	0.113732	0.120396	0.114463	0.112876	0.118638
γ lon, deg	0.185544	0.201937	0.187090	0.185893	0.186781	0.186811	0.187749	0.186306	0.186207	0.187255

Table 7.4: Average prediction evaluation 10-fold cross-validation

variable	**RMSE**, deg	**MAE**, deg
ϕ lat	0.154412	0.116056
γ lon	0.254895	0.188157

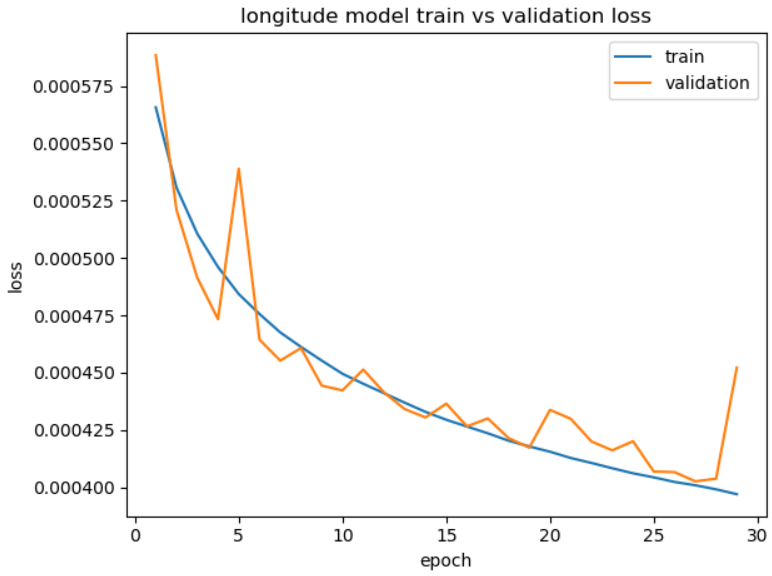


Figure 7.1: Fold 2 longitude loss functions 10-fold cross-validation

7.2. LSTM Prediction Analysis

Although it is stated in the previous section that the prediction accuracy could be improved, it seems that flight trajectory data might not be suitable as a dependent variable in LSTM prediction. From Fig. 7.2, encircled in green, it can be seen that the smoothness of certain trajectories are not persevered. It is unsure if this is due to lack of capturing the smoothness pattern or the model not being trained long enough. The same can be observed for the longitudinal sequences. Consequently one might try training the model longer and with a smaller learning rate. Another solution would be to apply smoothing filters to the predictions. Multiple smoothing and curve fitting algorithms for smoothing aircraft trajectories are presented by Tiancheng et al. [54]. However this is out of the scope of this study.

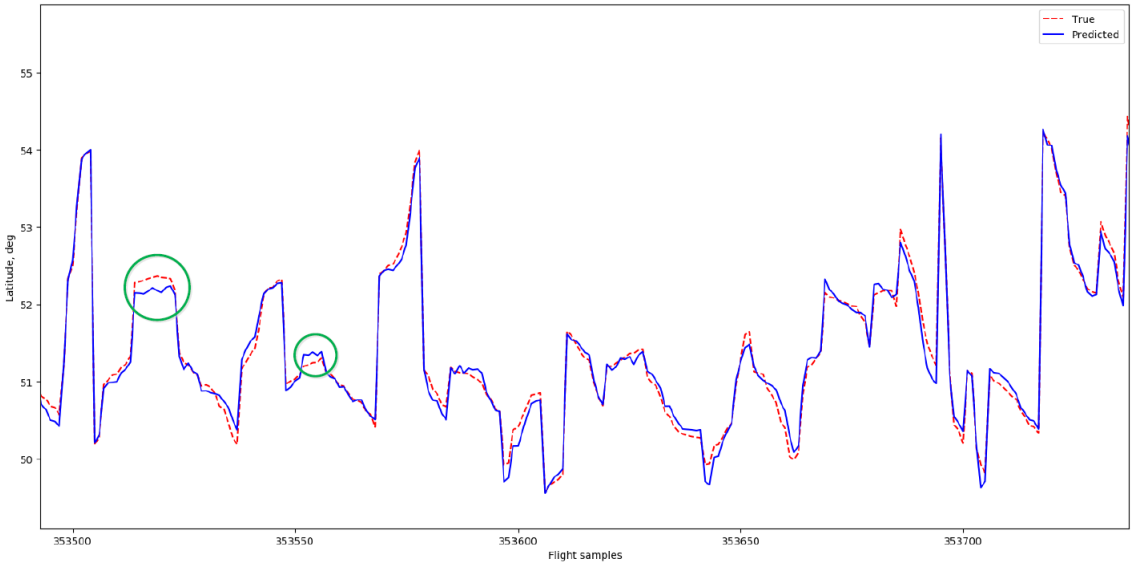


Figure 7.2: LSTM predicted (blue) -and true (red) latitude, where the jumps represent different flights

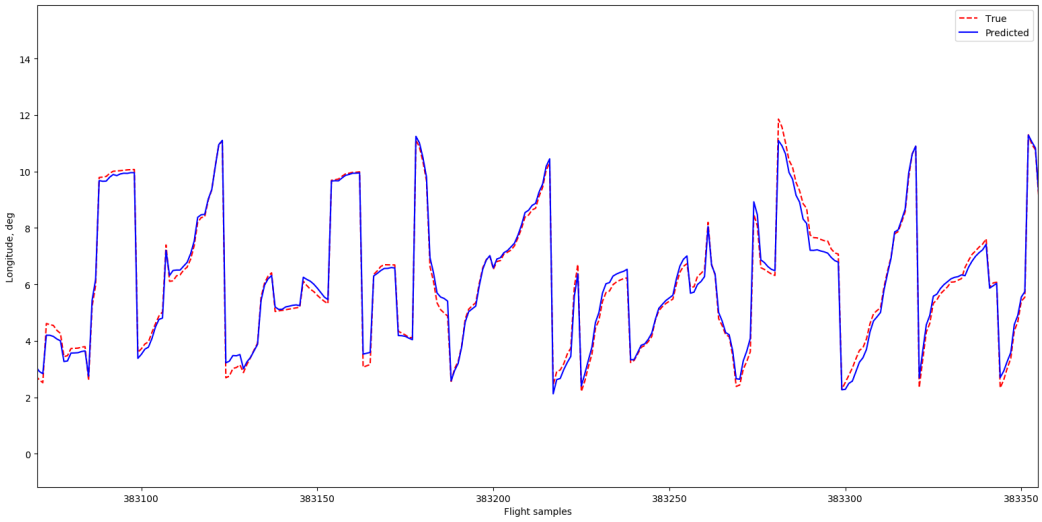


Figure 7.3: LSTM predicted (blue) -and true (red) longitude, where the jumps represent different flights

7.3. LSTM Based Deviation Analysis

With the final model trained, a comparison between deviations based on LSTM predictions versus FPLs can be made. In order to do this, 2 days of flights are kept as deployment test data. 7,860 flights passing through MUA on Friday 14th October, 2016 and Monday 27th June, 2016 are selected. The model has not seen these flights during training or validation. The test data is pre-processed using the statistical properties of the training data. The error prediction accuracy of the LSTM model on the test data is presented in table 7.5. As can be seen, both prediction measures are well within the range of the 10-fold cross-validations performance. Looking at the table on the right, it can be seen how planned trajectories perform on the same set of data. The accuracy prediction metrics of the planned routes w.r.t. the flown trajectories, presented in Tab. 7.6, show less accurate results when compared to the LSTM model.

Table 7.5: LSTM error performance on test data

variable	**RMSE**, deg	**MAE**, deg
ϕ lat	0.159819	0.116924
γ lon	0.253768	0.186139

Table 7.6: FPL error performance on test data

variable	**RMSE**, deg	**MAE**, deg
ϕ lat	0.200634	0.144633
γ lon	0.313652	0.234804

Due to the analysis presented in section 7.2, it is chosen to compare the deviation behavior on a subset of the test data. A sample of flights strictly complying to monotonic profiles for both LSTM and FPL predictions are selected. Although these profiles might be easier to predict, it will better reflect the difference in model performance.

The ATE statistics based on LSTM and FPL predictions with respect to flown trajectory are presented in tables 7.8 and 7.7 respectively. The deviations are labeled with the same categories as defined in section 2.5.

Here it can be seen that compared to the FPLs, the percentage of flights experiencing inherent longitudinal deviation slightly increased for LSTM predictions. The percentage of on-track minor deviations stayed approximately the same, whereas the on-track major deviation shows a significant improvement. Due to the filtering of off-track deviations, both models are bound to inherent and on-track deviations. Note that during deployment, some aircraft will still experience off-track major deviations.

Table 7.7: Statistics on flown trajectory maximum ATE w.r.t. FPLs of test data in MUA

Category	**Limits**	**Number of flights**	**% of total flights**
Inherent longitudinal deviation	$ATE <= 10\ nmi$	1,381	46.8
On-track minor longitudinal deviation	$10\ nmi < ATE <= 25\ nmi$	1,225	41.4
On-track major longitudinal deviation	$25\ nmi < ATE <= 75\ nmi$	348	11.8
Off-track minor longitudinal deviation	$75\ nmi < ATE <= 150\ nmi$	0	0
Off-track major longitudinal deviation	$ATE > 150\ nmi$	0	0

Table 7.8: Statistics on flown trajectory maximum ATE w.r.t. LSMT predictions of the test data in MUA

Category	**Limits**	**Number of flights**	**% of total flights**
Inherent longitudinal deviation	$ATE <= 10\ nmi$	1,573	53.2
On-track minor longitudinal deviation	$10\ nmi < ATE <= 25\ nmi$	1,196	40.5
On-track major longitudinal deviation	$25\ nmi < ATE <= 75\ nmi$	185	6.3
Off-track minor longitudinal deviation	$75\ nmi < ATE <= 150\ nmi$	0	0.0
Off-track major longitudinal deviation	$ATE > 150\ nmi$	0	0.0

Tables 7.9 and 7.10 show the CTE distributions. As can be seen, the number of flights deviating more than $5nmi$ has considerably decreased compared to the distribution obtained by the FPL. Consequently, the number of inherent deviated flights has increased. Similar to the ATE, there are no flights experiencing off-track deviations. Figures 7.4 and 7.5 give a better visual representation of the statistics.

Table 7.9: Statistics on flown trajectory maximum CTE w.r.t. FPLs of the test data in MUA

Category	**Limits**	**Number of flights**	**% of total flights**
Inherent lateral deviation	$CTE <= 5\ nmi$	1,864	63.1
On-track minor lateral deviation	$5\ nmi < CTE <= 20\ nmi$	853	28.9
On-track major lateral deviation	$20\ nmi < CTE <= 50\ nmi$	237	8.0
Off-track minor lateral deviation	$50\ nmi < CTE <= 100\ nmi$	0	0
Off-track major lateral deviation	$CTE > 100\ nmi$	0	0.0

From the analysis above, it can been concluded that the structural LSTM model tries to squeeze the spread of deviations towards inherent deviations, whereby improving the predictability of flights that experience major deviation w.r.t. the FPL en-route. The distributions of ATEs and CTEs are displayed in figures 7.6 and 7.7 respectively. As can be seen, the LSTM model decreases the spread from minor and major deviations

Table 7.10: Statistics on flown trajectory maximum CTE w.r.t. LSMT predictions of test data in MUA

Category	**Limits**	**Number of flights**	**% of total flights**
Inherent lateral deviation	$CTE <= 5\ nmi$	2,304	78.0
On-track minor lateral deviation	$5\ nmi < CTE <= 20\ nmi$	588	19.9
On-track major lateral deviation	$20\ nmi < CTE <= 50\ nmi$	62	2.1
Off-track minor lateral deviation	$50\ nmi < CTE <= 100\ nmi$	0	0.0
Off-track major lateral deviation	$CTE > 100\ nmi$	0	0.0

towards the center of the distribution. Furthermore, the spike present at 0 nmi in the FPL based distribution has disappeared in the LSTM distribution, as seen in Fig. 7.7. This indicates that LSTM predictions are less accurate than FPL based routes for absolute CTE inherent deviations less than 2 nmi. The mean and standard deviation for both TPEs and both models are presented in Tab. 7.11.

Table 7.11: Mean and standard deviation for both trajectory prediction errors and both model

	ATE FPL	ATE LSTM	CTE FPL	CTE LSTM
Mean, nmi	3.702866	0.778647	-0.186999	-0.104534
Standard deviations, nmi	12.830792	10.618904	7.230910	5.426019

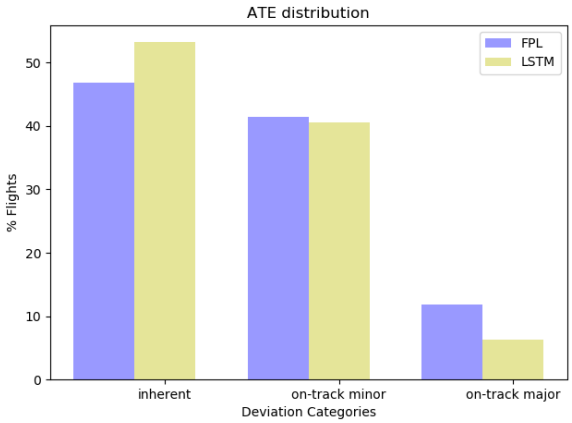


Figure 7.4: ATE bar chart

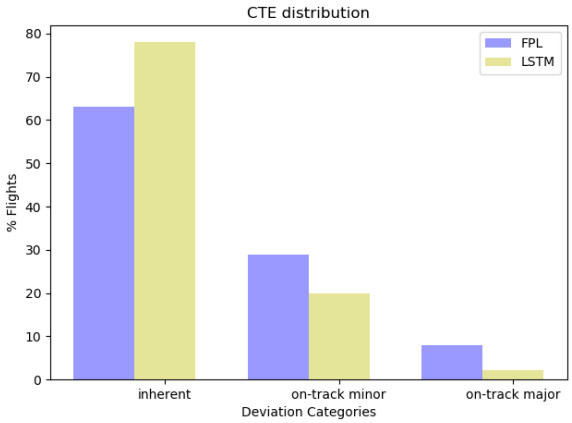


Figure 7.5: CTE bar chart

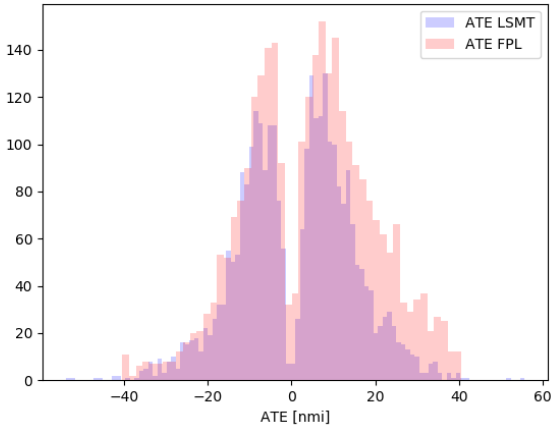


Figure 7.6: Test data ATE distribution

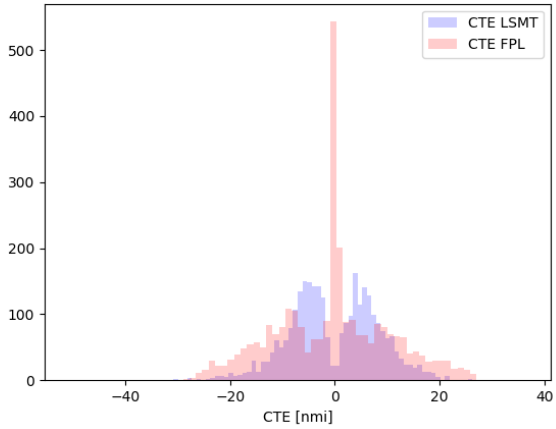


Figure 7.7: Test data CTE distribution

Finally, Figs. 7.8 and 7.9 visualizes the predictions and flown flights. In the first figure, although the LSTM CTE profile matches the flown track, the ATE appears to be different. In the second figure, it can be seen that the airspace entry point of the predicted LSTM path is not in agreement with the flown trajectory, in contrast to the FPL.

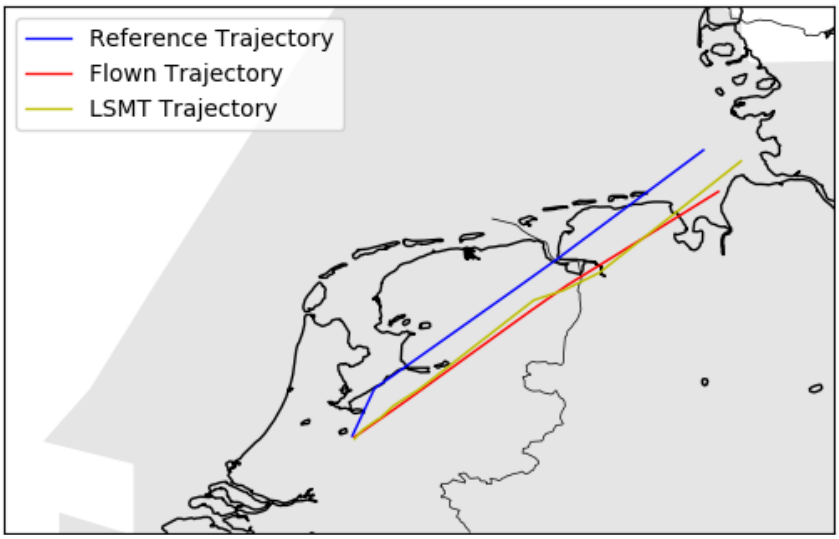


Figure 7.8: FPL vs flown vs LSTM of flight within MUA

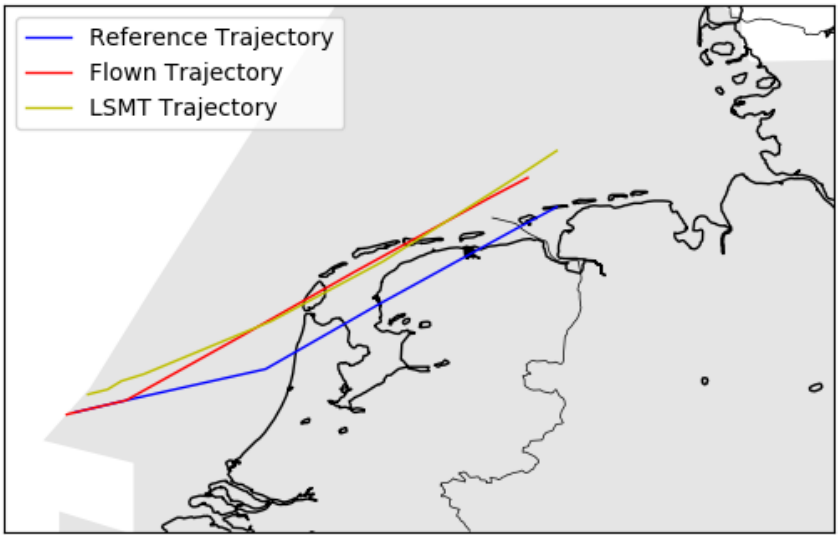


Figure 7.9: FPL vs flown vs LSTM of flight, additionally illustrating airspace sector entry points

8

Post-Analysis

With improved predictability, present and future ATM systems can also be improved. In this Chapter, a discussion is performed on the potential application of the LSTM model in ATFCM. The chapter starts by addressing the possible improvements in ATFM and subsequently demonstrates how the model can be used to improve the balance between air traffic demand and airspace capacity. Furthermore, it presents the *Aircraft Trajectory Deviation Feature Map*, which represents a map of features relating to aircraft deviation from planned routes, which can be explored and exploited in future work.

8.1. Application in Air traffic Flow Management

This study started off by presenting the inaccuracies in trajectory predictions based on the FPL in Chapter 2. As seen in comparison with the planned trajectory of the NM, it is often inaccurate w.r.t. the true trajectory. Additionally, all stakeholders have a different perspective of how a future trajectory, based on the FPL, will evolve as shown in Fig. 8.1. These limitations result in an incoherent collaboration, inefficient operations and subtle dissatisfaction between stakeholders.

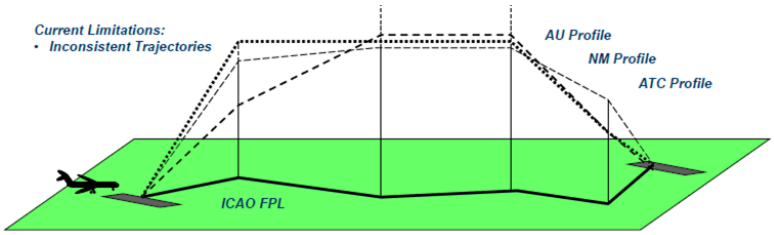


Figure 8.1: Representation of the limitations in aircraft trajectory predictions by all airspace stakeholders

A solution would be to provide a more accurate prediction as the reference trajectory. This can be accomplished by using the LSTM model as a last iteration by the NM, before sending out intent data to ANSPs. Two approaches are possible here;

- The NM retrieves new waypoints, in case of large deviations from FPL, after employing the LSTM model in a last trajectory prediction iteration and subsequently updates the FPL before dispatch to aircraft operator and ATC.
- All trajectory predictions are replaced by LSTM predictions.

ATC can therefore choose to either use the LSTM predictions to ensure safe flight or alternatively use the updated FPLs in its TPs to improve accuracy. Aircraft operator can therefore plan its operations based on the updated FPL.

8.2. Application in Air Traffic Demand Optimization

Predicted air traffic demand and available ATC capacity are determined by ANSPs, using tools such as the iFMP, that accesses local sector loads. Hereafter, appropriate sectorization can be employed during operations. Where traffic demand exceeds airspace capacity, ATFCM methods such as regulations, rerouting, departure slots, level-capping, etc. are employed by the NM upon request of ATC as a protective measure against overload.

Despite all these precautionary measures, excess aircraft do enter these restricted airspace after secondary deviation from their FPL. This is a result of:

- aircraft flying at a different flight level than the requested flight level.
- a change in estimated off-block time or calculated take-off time whereby aircraft departs at a different time.
- aircraft deviating from the FPL based planned trajectory

The These uncertainties lead to inaccurate airspace capacity estimations.

An alternative way to protect ATC from overload, is to employ the LSTM model to help improve the balance between air traffic demand and ATC airspace capacity. In order to access this, an evaluation is done on MUA local sector loads, based on the resulting monotonic flown trajectories. This is then compared to the loads generated by the planned trajectories and the LSTM predictions.

Occupancy count is the metric used in this study to access the local airspace capacity. Overload occurs when ATCo handles more aircraft than considered safe. In Fig. 8.2, the threshold value for this is depicted by the purple line. In the same figure, the red bars exceeding the threshold value are defined as sector *overload*. The green bars indicate an airspace sector capacity in equal balance with its air traffic demand and the blue bars are identified as *underload*. The counts are performed over a period of time. For simplicity, this analysis will cluster all monotonic flights in its respective sector. Hence, the time period can be seen as 2 days. Note that it is not the objective to discuss if occupancy count is the right metric to use, since other ATC workload related complexity metrics can also be used to estimate airspace capacity. It is mainly used as a tool to demonstrate the possible application of the LSTM model to airspace demand optimization.

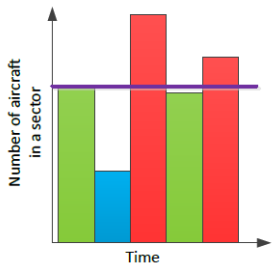


Figure 8.2: Visual representation of airspace sector occupancy count with threshold value indicated by blue line. The green bars represent a sector load in equilibrium with air traffic demand, while the red and blue bars indicate overload and underload respectively

A regulated sector exceeding its capacity by more than 10% is referred to as ATFCM *over-delivery* [55]. In sector management, this means that the number of aircraft entering a regulated sector exceeds the declared rate of the ATFM regulation. Thus for comparing the sector load state generated by the true trajectory to the sector load state based on the FPL and LSTM, the threshold for over -and underload is chosen as 10% margin of the reference sector counts.

When a comparison is made between true sector loads and estimated reference loads based on the FPL or LSTM, this means the following:

- Equal: true sector load is within 10% margin of the reference sector load
- Underload: true sector load is less than 90% of the reference sector load
- Overload: true sector load is more than 110% of the reference sector load

As can be seen in Fig. 8.3, the planned trajectories perform a poor job in estimating the true capacity for certain sectors. W.r.t. the planned flights, *Holstein* is in overload, while *Delta* and *Ruhr* are in underload. Compared to the sector loads estimation based on the LSTM prediction on the right, all elementary sectors are in equal balance with the predicted traffic demand. Tabs 8.1 and 8.2 present the load values of each sector in MUA, where 100% means that the true capacity is equal to the reference capacity.

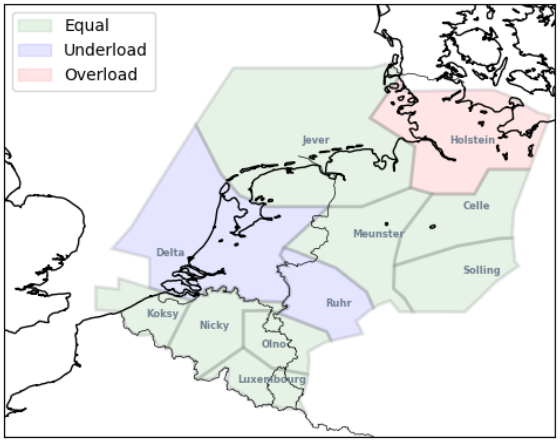


Figure 8.3: MUA sector load states generated by flown trajectories w.r.t. planned trajectories

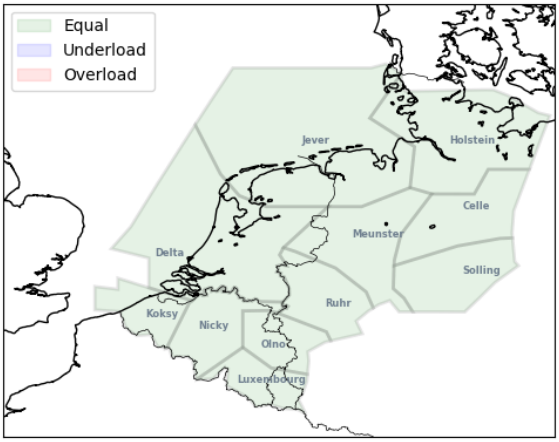


Figure 8.4: MUA sector loads states generated by flown trajectories w.r.t. LSTM predictions

Table 8.1: Percentage of true sector load with FPL based sector load as reference

sector	Celle	Delta	Holstein	Jever	Koksy	Luxembourg	Meunster	Nicky	Olno	Ruhr	Solling
% **load**	99	87	114	97	101	99	106	108	103	74	106

Table 8.2: Percentage of true sector load with LSTM based sector load as reference

sector	Celle	Delta	Holstein	Jever	Koksy	Luxembourg	Meunster	Nicky	Olno	Ruhr	Solling
% **load**	98	98	100	98	101	91	102	96	106	99	107

Hence, when estimating sector loads during tactical ATFCM, the LSTM predictions can be used as reference of what the true trajectories will be. The estimation of sector load state based on the LSTM, with FPL counts as reference, can be shown by Fig. 8.4. It can be seen that most sector load states are identified correctly. The only misclassified sector is *Nicky*, which has now become overload compared to the true sector load. A closer look at Table 8.3 shows the sector loads based on the LSTM model w.r.t. the FPLs. *Nicky* is only %1 above the threshold margin.

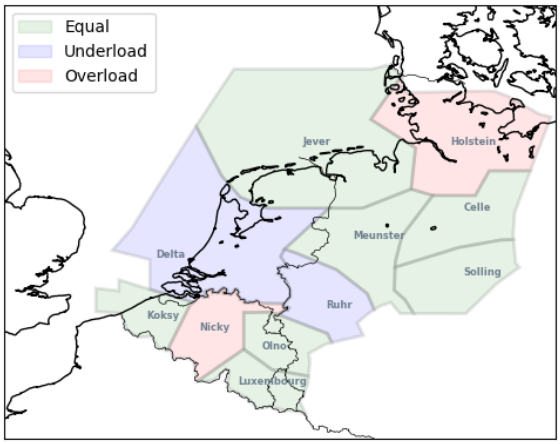


Figure 8.5: MUA sector loads generated by planned trajectories w.r.t. LSTM predictions

Table 8.3: Percentage of LSTM based sector load with FPL based sector load as reference

sector	Celle	Delta	Holstein	Jever	Koksy	Luxembourg	Meunster	Nicky	Olno	Ruhr	Solling
% **load**	102	89	114	100	100	108	104	111	97	75	101

From Fig. 8.5, it can be chosen to split *holstein* into 2 sectors. Moreover, in regards with balancing demand and airspace capacity, a decision can also be made to permit more air traffic passing through *Delta* and *Ruhr* and or reject FPLs that will eventually deviate and pass through *Holstein*.

8.3. Insights on Aircraft Deviation Related Features

Taking into account all sources of deviations, enables designing high-fidelity data-driven TP models, which learns from all trajectory based operational scenarios that impacts change in trajectory. However, the bottleneck is collecting data that is related to these factors from diverse data sources. Operational data among stakeholders are not openly shared. This information consists of data from service providers (NM, ANSP, Airport) and data from Airspace Users AU (e.g. aircraft operator). Even when the data are allocated for research, it is mostly applied to specific operational focus. In terms of data processing, the dissimilarity among the data can also be a challenge. For example, convective weather information from charts requires different processing than planned trajectory data.

Furthermore, the quality of the data poses another challenge. The FPL may be amended, but the modification might not reflect all updates. This negatively affects the fidelity of the model in absent of appropriate measures. Therefore, combining and contrasting various data from multiple data sources is advisable. Finally, the correct implementation of the disparate data set has to be implemented in suitable large scale data-driven models.

Current trajectory prediction models used by service providers, including the network manager, are model-based trajectory predictors. As shown in Fig. 1.1, these models always return the same trajectory for identical input. The model-based trajectory predictors do not consider actual and or past data. Furthermore, knowledge of aircraft operational settings, such as: initial aircraft weight, flight management system modes, etc., are required for these models. However, these settings remain unknown to stakeholders and research institutes. In the wake of innovation, state-of-the-art data-driven trajectory predictors have been known to achieve better prediction accuracy [25]. However, some research do not directly take into consideration factors causing aircraft to deviate from planned trajectories, while all research do not consider certain imperative contextual factors relating to complex traffic conditions and change in economic route. As a result, data-driven models lack relevant information for predicting highly accurate deviated routes.

Shi et al (2018) [26] employed aircraft dynamic states as inputs to a sliding window LSTM model to predict trajectories. Although these are relevant features for trajectory prediction, the inputs do not convey any specific information regarding the possible causes of deviations en-route. The parameters consisted of the following states:

- Position
- Speed
- Heading

Naessens et al (2017) [25]. had access to military aircraft flight data and was able to combine airspace structure -and availability of restricted airspace related features in predicting routes. However, the research did not take into consideration other factors such as atmospheric conditions or convective weather data. The following inputs to Naessens' proposed multilayered perceptron are:

- Entry coordination point, Exit Coordination point, After-Boundary Exit point, Entry flight level, Requested flight level and Exit flight level retrieved from the initial flight plan.
- Coordinates of the destination airport
- Compass bearing between the Entry coordination point and the departure airport
- Day of the week
- The half hour interval the flight is expected to enter the area of responsibility
- Reservation of military areas

Ayhan et al. (2016) [27] enriched their data set by combining airspace, surveillance -and weather related data. Ayhan et al. modeled a trajectory as a sequence of 4 dimensional spatio-temporal data cubes. The researchers trained a Hidden Markov Model to compute the maximal probability that generated the most likely trajectory points, compared to the ground truth. The features defining the spacial cube are:

- Reference coordinates (latitude, longitude, altitude)
- Wind speed
- Wind direction
- Air temperature

Liu et al.(2018) [28] also enriched their data set by combining airspace, surveillance, atmospheric conditions -and convective weather related data. Similar to Ayhan et al., Lui et al. definded a feature cube around a trajectory point that contained weather information in that cube. The features defining the cube are:

- Planned trajectory coordinates
- Wind speed
- Air temperature
- Convective weather polygon areas

8.3.1. Aircraft Trajectory Deviation Feature Map

TPs notably form the foundation of ATM systems and therefore, their accuracy is vital to improving the system as a whole. In order for TP models to have a complete picture of operations during trajectory execution, all sources of deviations should be exploited. However, it is understandably difficult to obtain such information. Hence, like in this study, other means should be evaluated when necessary. In this study, deviation related statistical parameters, proposed by [6], and other miscellaneous model regressors were added to atmospheric conditions, airspace structure, change in economic route and surveillance information to predict routes.

However, this is only the start of identifying the general issue and taking an initial step towards obtaining a high fidelity data-driven trajectory prediction model. Therefore, the *Aircraft Trajectory Deviation Related Feature Map* is proposed in Tab. 8.4. The map provides an overview of which deviation categories that still need be explored, in order to improve the predictability of TPs. Considering most features in this map relate to air traffic conditions, it is most important to initially collect a complete data set.

Quantitative horizontal TPEs are assigned positive or negative sign, which indicates if aircraft deviated to the right or to the left of the reference track for CTE, and alternatively behind or ahead of the reference trajectory point. This is also represented by the histogram distributions in Figs. 2.5 and 2.6. The main reason for this is trajectory operations ahead of the aircraft. For example, a downstream sector capacity might be high and therefore aircraft is rerouted. The direction of reroute might also depend on weather conditions. Suppose there is convective weather to the left of the sector in overload, aircraft will most likely be rerouted towards the right side. In this example, altitude traffic flow measures are not considered. Based on this insight, *radial features* are proposed. These type of contextual features will provide models additional patterns to uncover in predicting trajectories. Fig. 8.6 demonstrates what is meant by a radial feature. In this Fig., an arc is defined by a radius R ahead of the aircraft. R can be an average distance error between planned -and true trajectories. Weather conditions are sampled at the location of the aircraft and on specific points on the arc. Similarly, instead of only sampling traffic conditions in the aircraft's current sector, traffic conditions in neighbouring sectors ahead of the aircraft are also obtained. The main difference in this approach, compared to the 4 dimensional spatio-temporal cube approach, proposed by [27, 28], is that conditions behind the aircraft do not affect the prediction of the next trajectory point. This type of feature sampling can also be applied to various deviation related features. In the feature map, this is highlighted by a yellow cell.

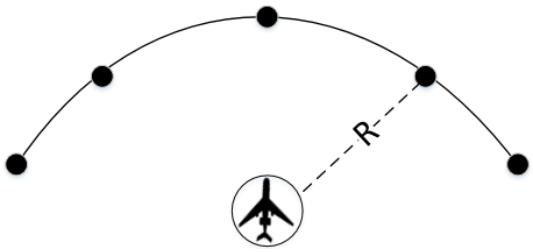


Figure 8.6: Representation of radial feature sampling approach

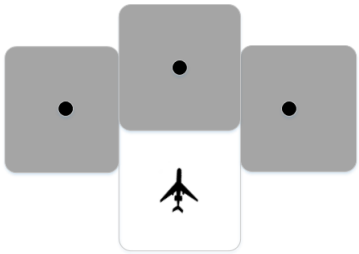


Figure 8.7: Representation of airspace sector radial feature sampling approach

Table 8.4: Aircraft trajectory deviation related feature map

Category	Feature	Eq./Descprition	Source (defined)
Atmospheric conditions	• Wind Direction • Wind speed • Air Temperature	N.A.	This study
Complex traffic conditions	Occupancy count	N.A.	N.A.
Complex traffic conditions	Relative speed deviation	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(v_i - v_m)}$ (8.1) Where N is number of flights, v_i is speed of flight i, v_m is mean of flight speeds and σ the standard deviation	[48]
Complex traffic Conditions	Sector Density	$Dens(i) = 1 + \sum_{j=N,J\neq i}^{N} e^{-\alpha \frac{\left\|\bar{d_{ij}}\right\|}{R}}$ (8.2) Where N is number of aircraft in a given sector. $\left\|\bar{d_{ij}}\right\|$ is the norm distance vector of aircraft i and j. α is a weighted coefficient which according to the source can remain 1. R is the radius representing the contextual network. It can be set as an average distance error between planned trajectories and true tracks.	[48]
Complex traffic Conditions	Traffic Convergence	$Conv(i) = \sum_{j=N,J\neq i}^{N} \left\lvert \frac{d\left\|\bar{d_{ij}}\right\|}{dt} \right\rvert 1_{R^-} \left\{ \frac{d\left\|\bar{d_{ij}}\right\|}{dt} \right\} e^{-\alpha \frac{\left\|\bar{d_{ij}}\right\|}{R}}$ (8.3) $Conv(i)$ suggest the traffic convergence w.r.t. the i^{th} aircraft, where 1_{R^-} is the negative real number indicator function. It allows for taking into account flights approaching aircraft i. For description of the rest of the variables, see sector density.	[48]
Complex traffic Conditions	Traffic Divergence	$Div(i) = \sum_{j=N,J\neq i}^{N} \left\lvert \frac{d\left\|\bar{d_{ij}}\right\|}{dt} \right\rvert 1_{R^+} \left\{ \frac{d\left\|\bar{d_{ij}}\right\|}{dt} \right\} e^{-\alpha \frac{\left\|\bar{d_{ij}}\right\|}{R}}$ (8.4) $Div(i)$ suggest the traffic divergence w.r.t. the i^{th} aircraft, where 1_{R^+} is the negative real number indicator function. It allows for taking into account flights flying away from aircraft i. For description of the rest of the variables, see sector density.	[48]
Conflicting traffic	Predicted conflicts	This feature is estimated by determining which aircraft fly on the same FL and on intersecting routes. Determine which of aircraft reach intersection point in a relatively short time and with a short time difference w.r.t. each other.	[48]
Change in economic route	Route charges	Charges incurred en-route along different airspace	Allft+
Change in economic route	Route efficiency	Planned route length over great circle length	[6]
Change in economic route	Fuel consumption	Total fuel consumption per total planned route length	This report

9
Conclusion

Based on statistics performed on 1,388,338 flights passing through European Civil Aviation Conference airspace between February and December 2016, there is evidence of large deviations from filed flight plans. Globally, approximately 32% of all flights experienced on-track major lateral deviations between 20 - 50 nautical miles, while 35% experienced on-track major longitudinal deviations ranging from 25 - 75 nautical miles. These type deviations are not accounted for during flight planning and therefore disrupting air traffic operations in flight. This causes air traffic control decision support tools, which depend on trajectory predictors, to be unreliable. As a result, it is less feasible to reduce air traffic control operator's mental workload and thus increase sector capacity. Consequently, air traffic predictability needs to be improved. Quantitatively, this means reducing the measure of spread of aircraft deviations from filed flight plans. Therefore, the proposed study's objective is to predict trajectories before departure by applying machine learning techniques and to quantify and analyze their deviations w.r.t. the true trajectories within Maastricht upper airspace. With the insights obtained, proposals are made on how to improve current and future air traffic flow and capacity management systems.

In order to accomplish this, planned air traffic management data is employed in a long short term memory (LSTM) model. Included in this data are aircraft deviation related statistical parameters and other miscellaneous deviation related independent variables that have only now been explored in predicting trajectories. This attempt has been made as an initial step to providing more deviation related contextual features to data-driven trajectory predictors.

It has been shown that the LSTM trajectory predictor outperforms the current model-based trajectory predictor, generating planned routes based on filed flight plans for flights in Maastricht upper airspace. The root mean squared error of the LSTM latitude predictions is 0.159819 deg, whereas the planned trajectories have a root mean squared error of 0.200634 deg w.r.t. to the true trajectories. The obtained LSTM prediction error for the longitude dependent variable is 0.253768 and 0.313652 for the planned trajectories. Dispite the error improvement, certain LSTM predictions were not smooth trajectories. Therefore, a subset of strictly monotonic trajectories were analyzed for deviations. Compared to the planned trajectories, the cross-track -and along-track error (CTE, CTE) along the LSTM predicted routes improved. The respective mean and standard deviation for CTE for both FPL -and LSTM based routes are: $-0.186999\ nmi$, $7.230910\ nmi$ for FPL and $-0.104534\ nmi$, $5.426019 nmi$ for LSTM. As a result, the distribution of deviations shifted towards the center, whereby reducing the spread and thus improving the pre-departure predictability of monotonic aircraft trajectories in Maastricht upper airspace.

Because all stakeholders have different trajectory predictors, resulting in different trajectory profiles, when distributing updated filed flight plans based on LSTM predictions, the quality of the input data of those filed flight plans will be improved, and thus provide more accurate trajectory predictions. This improves air traffic predictability for all stakeholders. Airline operators can plan their operations according to what is expected in real life, and therefore reduce inefficiencies and costs. Similarly, air traffic control can plan their airspace for longer look-ahead times in real time flow management services.

Also, off-line air traffic management services will benefit from the resources obtained from this study, for instance, in optimizing air traffic demand. The study shows that it is possible to obtain more accurately predicted traffic counts for elementary airspace in Maastricht upper airspace. As a result, better decisions regarding sector overload and underload can be taken.

10
Recommendations

Several aspects are proposed in this chapter to improve the LSTM model. These cover the topics of feature engineering, data preprocessing, model design, model training and postprocessing of the predictions.

1. Feature engineering

    - As mentioned in the post-analysis, more contextual features are required to provide data-driven trajectory predictors a complete picture of trajectory based operations. Features relating to complex traffic conditions are still omitted in all trajectory prediction research. Therefore, the features proposed in the *Aircraft Trajectory Deviation Related Feature Map* in Chapter 8 can be added to the input space of the model.
    - This study also proposes radial feature sampling approach which provides network context to the model. The model will take into account deviation related factors, such as: complex traffic conditions, airspace capacity -and volume, weather, etc., which are occurring ahead of the aircraft's current position.
    - Even though principle components were used to train the model, a feature importance study can be carried out to see which feature contributes the most to predicting trajectories. This will provide more insights on the use of certain features.

2. Model design

    - Bi-directional LSTM model should be considered. Although it has seen great achievements in areas such as natural language processing, as far as research shows, this approach has yet to be applied to aircraft trajectory prediction. Unlike uni-directional LSTM, bi-directional LSTM, processes data in two directions, forward and in the reverse order. This allows the model to additionally capture future information. As a result, it is able to use past and future information to produce the current prediction [56]. This model will be extremely useful when more deviation related contextual features are added.

3. Model Training

    - From the loss trends, shown in chapter 7, it can be seen that the model has not finished learning. Training was stopped at this point due to the availability of computational resources and time. However, it is advisable to increase the number of epochs. From smaller test runs, the losses seems to plateau at approximately 50 epochs.
    - The fluctuations in the validation loss indicates a high learning rate. The learning rate can therefore be reduced. Moreover, (cyclic) learning rate schedules can be explored for further training.

4. Postprocessing of predictions

    - Despite the LSTM model's inability to generate smooth flights for certain cases, it is possible to improve those predictions by apply smoothing to trajectories where needed. Kalman smoothing, which is easily applicable to the unequally spaced time series, as shown by Choudhry et al. [57], can be employed.

Bibliography

[1] Eurocontrol. Ddr2 reference manual for general users 2.1.3. Available at: `http://www.eurocontrol.int/ddr` [Accessed 22-06-2017], .

[2] Eurocontrol. Saam. Available at: `http://www.eurocontrol.int/saam` [Accessed 1-11-2017], .

[3] T. H. Lee G. Cai, B. M. Chen. Coordinate systems and transformations. *In: Unmanned Rotorcraft Systems. Advances in Industrial Control.*, (2011).

[4] S. Yeung F.F. Li, J. Johnson. *Lecture 4: Backpropagation and Neural Networks.* cs231n: Convolutional Neural Networks for Visual Recognition (Stanford University). April 2017.

[5] J. Friedman T. Hastie, R. Tibshirani. *The Elements of Statistical Learning.* Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[6] R.N. Mantegna G. Gurtner F. Lillo C. Bongiorno, S. Micciche. Statistical characterization of deviations from flight plan trajectories in air traffic management. *Journal of Air Transport Management*, Vol. 58: 152–163, 2017.

[7] A. Courville I. Goodfellow, Y. Bengio. *Deep Learning.* MIT Press, November 2016.

[8] Eurocontrol. Eurocontrol seven-year forecast february 2018. Available at https://www.eurocontrol.int/, February 2018.

[9] EUROCONTROL. 2015 comparison of air traffic management-related operational performance: U.s./europe. Technical report, FAA/European Commission/EUROCONTROL, August 2016.

[10] SESAR Joint Undertaking. Sesar solutions catalogue. Publications Office of the European Union, 2016, November 2016.

[11] ICAO. Doc 4444, procedures for air navigation services — air traffic management. November 2016.

[12] A. Majumdar W. Y. Ochieng L. Martinez P. Hendrickx G. Tobaruela, W. Schuster. A method to estimate air traffic controller mental workload based on traffic clearances. *Journal of Air Transport Management*, Vol. 39(2014):59–71, May 2014.

[13] R. Christien G. Flynn, A. BenKour. Pessimistic sector capacity estimation. Eurocontrol, November 2003.

[14] D. McNally C. Gong. A methodology for automated trajectory prediction analysis. *AIAA Guidance, Navigation, and Control Conference*, AIAA 2004-4788:152–163, 16-19 August 2004.

[15] G. McDonald J. Bronsvoort M. Paglione, I. Bayraktutar. Lateral intent error's impact on aircraft prediction. *Journal of Air Traffic Control Quarterly*, Vol. 18(1):29–62, January 2010.

[16] R.J. Hansman T.G. Reynold. Investigating conformance monitoring issues in air traffic control using fault detection techniques. *Journal of Aircraft*, 42(5):1307–1317, September-October 2005.

[17] G. Fourlas and J. Lygeros. Detection of flight plan divergence in the horizontal plane. *AIAA Guidance, Navigation, and Control Conference*, (21–24), August 2006.

[18] G. Fourlas and J. Lygeros. Detection of aircraft divergence from its flight plan in the vertical plane. *46th IEEE Conference on Decision and Control*, (12–14), Dec 2007.

[19] H. Yang R. Wang H. Yan, B. Yang. Probabilistic approach to conformance monitoring using gaussian processes. *Journal of Guidance, Control, and Dynamics*, 40(6):1403–1414, June 2017.

[20] R.J. Hansman R. Jordan T. Reynolds H. Balakrishnan M.C.R. Murca, R. DeLaura. Conformance monitoring under uncertainty in trajectory. August 2012.

[21] J. Hoekstra J. Sun, J. Ellerbroek. Large-scale flight phase identification from ads-b data using machine learning methods. *7th International Conference on Research in Air Transportation (ICRAT)*, 2016.

[22] E. Hüllermeier. Does machine learning need fuzzy logic? *Journal of Fuzzy Sets and Systems*, Vol. 281: 292—-299, September 2015.

[23] R.J. Hansman R. Jordan T. Reynolds H. Balakrishnan M.C.R. Murca, R. DeLaura. Trajectory clustering and classification for characterization of air traffic flows. June 2016.

[24] E. Feron M. Gariel, A.N.Srivastava. Trajectory clustering and an application to airspace monitoring. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1511 – 1524, August 2011.

[25] M. Piatek K. Schippers R. Parys H. Naessens, T. Philip. Predicting flight routes with a deep neural network in the operational air traffic flow and capacity management system. *EUROCONTROL Maastricht Upper Area Control Centre*, July 2017.

[26] Q. Pan B. Yan H. Zhang Z. Shi, M. Xu. Lstm-based flight trajectory prediction. *2018 International Joint Conference on Neural Networks (IJCNN)*, July 2018.

[27] S. Hana A. Samet. Aircraft trajectory prediction made easy with predictive analytics. 10.1145/2939672.2939694, August 2016.

[28] M. Hansen Y. Liu. Predicting aircraft trajectories: A deep generative convolutional recurrent neural networks approach. arXiv:1812.11670, December 2018.

[29] Eurocontol. Integrated flow management position (ifmp). Available at: `http://www.eurocontrol.int/articles/integrated-flow-management-position` [Accessed 01-08-2017].

[30] FAA-EUROCONTROL COOPERATIVE RD. Action plan 16: Common trajectory prediction capability. common trajectory prediction-related terminology. n.d.

[31] S. Mondoloni. Aircraft trajectory prediction errors: Including a summary of error sources and data. July, 2006.

[32] R. Vivona D. Karr. Conflict detection using variable four-dimensional uncertainty bounds to control missed alerts. *AIAA Guidance, Navigation, and Control Conference*, AIAA 2006-6057, August 2006.

[33] C. L. Philip Chen L. Chen M. Gan, H.X. Li. A potential method for determining nonlinearity in wind data. 2(2):74 – 81, August 2015.

[34] J. Werner S. Braun E. Stroe-Kunold, T. Stadnytska. Estimating long-range dependence in time series: An evaluation of estimators implemented in r. *Behavior Research Methods*, Vol. 41(3):909–923, August 2009.

[35] G. Pujolle A. Azzouni. A long short-term memory recurrent neural network framework for network traffic matrix prediction. June 2017.

[36] F. Bradfield. Sky cast: A comparison of modern techniques for forecasting time series. January 2018.

[37] G. Chen. A gentle tutorial of recurrent neural network with error backpropagation. *CoRR*, abs/1610.02583, January 2018.

[38] K. Mohiuddin K.J. Anil, J. Mao. Artificial neural network: A tutorial. *IEE Computer Special Issue on Neural Computing*, 29(3):31–44, March 1996.

[39] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60:84–90, 2012.

[40] I. Sutskever. *TRAINING RECURRENT NEURAL NETWORKS*. 2013.

[41] J. Schmidhuber S. Hochreiter. Long short-term memory. 9(8):1735–1780, 1997.

[42] C. Bielza P. Larrañaga H. Borchani, G. Varando. A survey on multi-outputregression. *WIREs Data Mining Knowl Discovw*, Volume(5):216–233, July 2015.

[43] Dr ir J. Ellerbroek J. Rudnyk, Prof. dr ir J.M. Hoekstra. Trajectory prediction for medium term conflict detection. Available at: http://cs.lr.tudelft.nl/atm/projects/trajectory-prediction-for-medium-term-conflict-detection/ [Accessed 20-07-2017], March 2016.

[44] J. Malczewski. *GIS and Multicriteria Decision Analysis*. John Wiley and Sons, April 1999.

[45] T. Vincenty. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey Review*, Vol. XXIII(176):88–93, April 1975.

[46] Movable Type Ltd. Movable type scripts. calculate distance, bearing and more between latitude/longitude points. Available at: http://www.movable-type.co.uk/ [Accessed 01-10-2017].

[47] R. Ferdinand A. Q. Ho. Bilinear interpolation. Paper presented at the annual meeting of the Oklahoma Research Day, Cameron University, Lawton, November 2014.

[48] G. Szabó Számel, I. M udra. Applying airspace capacity estimation models to the airspace of hungary. *Period. Polytech. Transp. Eng. B.*, 43(3):120–128, January 2015.

[49] D. François T. Vincenty. The curse of dimensionality in data mining and time series prediction. *Computational Intelligence and Bioinspired Systems. IWANN 2005. Lecture Notes in Computer Science*, Vol. (3512):758–770, 2005.

[50] D. Delahaye Z. Wang, M. Liang. Short-term 4d trajectory prediction using machine learning methods. *SID 2017, 7th SESAR Innovation Days*, November 2017, Belgrade, Serbia.

[51] R. Cohn J. Russell. *Spearman's Rank Correlation Coefficient*. Book on Demand, 2012.

[52] J. Shlens. A tutorial on principal component analysis. *eprint arXiv:1404.1100*, April 2014.

[53] C. Ying Q. V. Le S. L. Smith, P. Kindermans. Don't decay the learning rate, increase the batch size. Published as a conference paper at ICLR 2018.

[54] S. Sun J. M. Corchado T. Li, H. Chen. Joint smoothing, tracking, and forecasting based on continuous-time target trajectory fitting. *Journal of Forecasting*, 99, August 2017.

[55] Skybrary. Sector over-deliveries and overloads. Available at: https://www.skybrary.aero/index.php/Sector_Over-deliveries_and_Overloads [Accessed 15-09-2019].

[56] S. Poria E. Cambria T. Young, D. Hazarika. Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, Vol. 13(3):55–753, August 2018.

[57] W. Hao C. Taufiq. Forecasting ability of garch vs kalman filter method: evidence from daily uk time-varying beta. *IEEE Transactions on Automation Science and Engineering*, 27(8):670–689, August 2017.