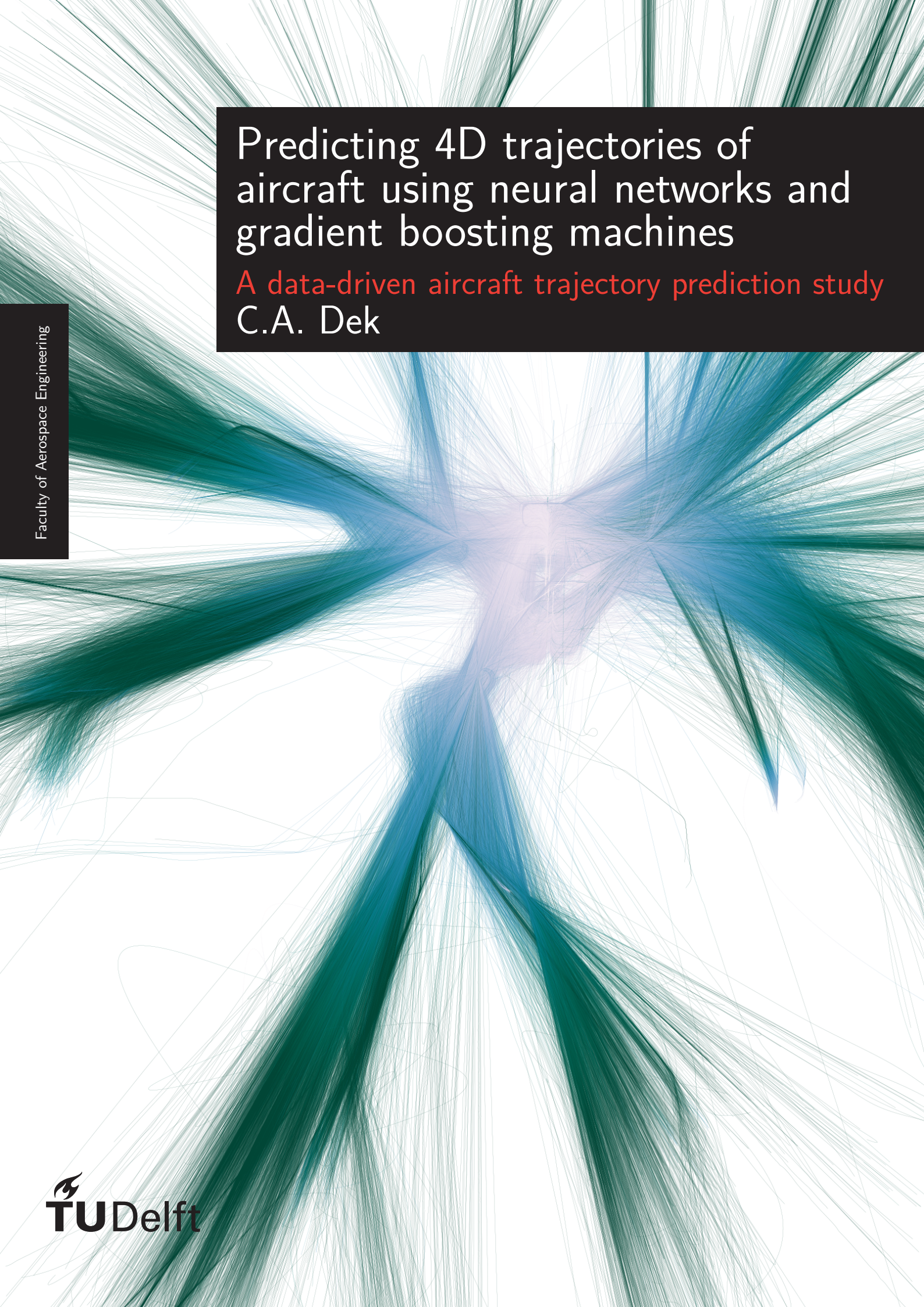
Predicting 4D trajectories of aircraft using neural networks and gradient boosting machines

A data-driven aircraft trajectory prediction study

C.A. Dek

Faculty of Aerospace Engineering



TU Delft

Predicting 4D trajectories of aircraft using neural networks and gradient boosting machines

A data-driven aircraft trajectory prediction study

by

C.A. Dek

to obtain the degree of Master of Science

at the Delft University of Technology,

Student number:	4300653	
Readers:	Prof. Dr. Ir. J. M. Hoekstra,	TU Delft, supervisor
	Dr. Ir. J. Ellerbroek,	TU Delft, supervisor
	Dr. O.A. Sharpanskykh,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

The image on the front page shows all trajectories taken into account in this study. Schiphol Airport can be found the middle where all trajectories converge and the trajectories are colored based on their altitude.

TU Delft

Acknowledgements

I would like to express my gratitude towards my supervisors Jacco Hoekstra and Joost Ellerbroek. Their valuable feedback and critical questions led to many new insights and improved the quality of my work. Especially, I would like to thank them for providing the space to explore the problems by myself and letting me find my own way to solve them. Also, I would like to thank Bart, Sjef and Thomas for reading and providing feedback on this thesis report.

A word of appreciation goes out to my friends and fellow students with whom I shared a lot of positive and some lesser moments alike throughout my studies, extracurricular activities and my thesis in particular. And to my roommates, with whom I spent the last months of my thesis during the lockdown of the Netherlands and who had to listen to my never-ending stories on aircraft trajectories.

Most importantly, I would like to thank my family for supporting me unconditionally both in and outside my studies. Whether I was around or not, it feels great to have a place called home to be able to fall back to when needed.

> *"Give me six hours to chop down a tree and I will spend the first four sharpening the axe"*
> — Abraham Lincoln

C.A. Dek
Delft, June 2020

Contents

List of Acronyms

ADAM	Adaptive Moment estimation
ADS-B	Automatic Dependent Surveillance - Broadcast
ALT	Altitude
APM	Aircraft Performance Model
APP	Aircraft Performance Parameter
ATC	Air Traffic Control
ATCo	Air Traffic Controller
ATM	Air Traffic Management
BADA	Base of Aircraft Data
CAS	Calibrated Airspeed
DBSCAN	Density Based Spatial Clustering for Applications with Noise
DT	Decision Tree
ECMWF	European Centre for Medium-Range Weather Forecasts
EHAM	Amsterdam Schiphol Airport
ERA	ECMWF Re-Analysis
FIR	Flight Information Region
GBM	Gradient Boosting Machines
GFS	Global Forecasting System
IAS	Indicated Airspeed
k-NN	k-Nearest Neighbours
LAT	Latitude
LON	Longitude
LSTM	Long Short-Term Memory
LVNL	Luchtverkeersleiding Nederland (ATC Netherlands)
ML	Machine Learning
MLP	Multilayer Perceptron
MSE	Mean Squared Error
NOAA	National Oceanic and Atmospheric Administration
OHE	One-Hot Encoding
PCA	Principal Component Analysis
RMSE	Root Mean Squared Error
ROC	Rate of Climb
SSPD	Symmetrized Segment-Path Distance
STAR	Standard Arrival Route
TAS	True Airspeed
TEM	Total Energy Model
TID	Trajectory ID
TP	Trajectory Prediction

List of Figures

List of Tables

I

Scientific Paper

Predicting 4D Trajectories of Aircraft using Neural Networks and Gradient Boosting Machines

C. A. Dek*, J. M. Hoekstra‡, J. Ellerbroek‡
Control and Simulation, Faculty of Aerospace Engineering
Delft University of Technology, Delft, The Netherlands

***Abstract*—Data-driven trajectory prediction is one of the key pillars of the future ATM system. Recent research focuses on using novel data sources and machine learning algorithms to improve the performance of 4D trajectory prediction, enabling safer and more efficient routing of aircraft. In this paper a framework for data sourcing and preparation for such predictors is presented, as well as a comparison of three of the best performing prediction algorithms from literature to a baseline method. Currently such comparisons are lacking, making it hard to determine which techniques provide the best results. Using an ADS-B antenna and various online data sources a trajectory set of 40,000 trajectories is built. Two clustering methods are tested and it is found that clustering trajectories using Density Based Clustering for Applications with Noise (DBSCAN) performs poorly on our data set of arriving flights. Too many trajectories are classified as outliers while DBSCAN is not capable of separating the trajectories in distinct clusters. A clustering method based on the STARs of the airport is proposed, which performs better in terms of accuracy and efficiency. Finally, a baseline simulation using Aircraft Performance Models is compared to a deep neural network, a Long Short-Term Memory (LSTM) network and to Gradient Boosting Machines (GBM) for trajectory prediction. It is found that the latter outperforms the other methods overall, while it was expected that predictors based on LSTMs would provide more accurate results. It is concluded that long-term dependencies in trajectory data, on which LSTMs perform well, are less important than categorical indicators, on which GBMs perform better, in trajectory prediction.**

***Index Terms*—4D trajectories, ADS-B, DBSCAN, deep neural networks, Gradient Boosting Machines, Long Short-Term Memory networks, machine learning, trajectory clustering, trajectory prediction**

I. Introduction

Data-driven trajectory prediction has been identified as one of the key pillars for the future Air Traffic Management (ATM) system by Eurocontrol and is being investigated in the DART project, as part of SESAR [1]. The incentive for this effort is the increasing amount of air traffic that is to be handled safely in a constrained airspace whilst simultaneously increasing individual flight efficiency.

The predictions aimed for are 4D trajectories, which consist of points in 3D space over time that together form the trajectory an aircraft executes. Currently, 4D trajectories are predicted using aircraft performance models (APMs), which use aircraft performance parameters (APPs), flight plans and meteorological conditions as inputs [2]. These APMs provide fairly accurate results, however do not take external influences into account. Besides, all common procedures in an airspace have to be known and provided to the APM. Using existing and novel data types, such as Automatic Dependent Surveillance - Broadcast (ADS-B) data and route information, patterns can be found that indicate aircraft and ATC behaviour in a certain situation, without needing prior knowledge of the airspace.

Past research has focused on identifying these patterns using various machine learning techniques, ranging from simple linear regression models to deep recurrent neural networks [3]. Also, the input parameters to such models have been evaluated extensively. However, most of these predictors are not tested against each other or the systems currently in place, making it difficult to determine which techniques provide the best results.

The research presented in this article consists of two parts, namely an investigation into the data sources and data preparation required for such predictors, as well as a comparison of the best performing predictors from literature set against a baseline method representative of the current system. The goal is to identify the best performing predictors along with the required pre-processing steps.

The structure of this article is as follows. First, an overview of the related work from literature will be provided in section II, along with a more in depth literature review of the machine learning algorithms used in this study. The methodology of the research is detailed in section III. The results are presented in section IV, after which a discussion on these results is provided in section V. Finally, conclusions drawn from this research are provided in section VI and recommendations for future research can be found in section VII.

II. Related work

As stated in the introduction, the work presented here consists of two parts. Therefore, first, related work on the influence of input parameters on predictors is provided to define the required input to the predictors. Also, an overview of efforts in the field of trajectory clustering and air traffic flow identification is presented, which is a second preparation step that can be used to optimise the performance of the predictors [4].

Secondly, the prediction phase of the research is supported by an overview of various prediction methods, starting with related work using APMs as this will serve as the baseline in

*MSc student, ‡Supervisor.

the comparison. Next, novel techniques predicting trajectories using machine learning are detailed and the predictors to be compared in this research are identified. From section II-E onwards, the machine learning algorithms used in this research are described in detail.

A. Relevant input parameters to trajectory predictors

Two major sources of data on aircraft trajectories exist. First of all, radar data can be used to reconstruct 4D trajectories, and is used both for predictors based on APMs as well as for data driven predictors [5], [6], [7], [8]. Secondly, ADS-B data is often used to reconstruct 4D trajectories [9], [10], [11], [12]. ADS-B data is broadcasted by commercial aircraft through Mode-S Extended Squitter (1090 MHz) and is publicly receivable by everyone with an antenna [13]. ADS-B data constitutes of a time designator, an aircraft identifier, altitude, lat / lon position, heading, speed and rate of climb. Also, ADS-B positions have been proven to be more accurate than radar positions [14]. Considering that it is openly accessible, accurate and comprises of the position data required to reconstruct the flown trajectories, it is chosen as the main data source in this research.

To determine which additional parameters have a significant influence on the results of trajectory prediction, several studies have been performed using APMs. From these studies it can be concluded that aircraft intent is a driver for the accuracy [15], [16], as well as CAS / Mach settings [17]. Additionally, the importance of aircraft weight has been shown to be one of the major factors in the quality of a predictor, especially for the vertical part of the prediction [15], [16], [18]. The accuracy in the horizontal part of a trajectory is shown to increase when wind conditions are taken into account [17] [18]. Similarly, flight duration errors are linked to the accuracy of the true airspeed [18], for which the wind conditions are of importance. The temperature of the air, however, was found not to provide significant advantages [17]. Although these studies were performed for predictors based on APMs instead of the data-driven predictors tested in this research, the influence of wind conditions and the aircraft weight are considered to have equal importance for this research and will therefore be used. Aircraft intent data is expected to be of more importance for predictors using APMs than for data-driven predictors as the latter can attempt to forecast the intent and routing themselves. Although it is noted that it could still be valuable information for any predictor, it is not included in this research.

B. Clustering methods for air traffic flows

The second step in preparing the data sets for the predictors is trajectory clustering. Clustering the trajectories in distinct and similar groups and training algorithms separately on these specific sets of data enables detecting more detailed patterns in the trajectory data [19]. Clustering can be based on known characteristics such as the aircraft type, route or origin of a flight, or can be done by using unsupervised machine learning methods that focus on the spatial properties of a trajectory.

Early work in this field focused on finding traffic flows and routes, which is a task comparable to clustering for prediction. A first attempt in this direction was performed using partitional, density based and hierarchical clustering [20], from which it was concluded that density based clustering performs well, but that the clustering process should be faster. Later, density based methods were compared to k-means clustering on sparse data, on which the latter performed better [21]. However, when larger data sets of over 1,000 trajectories are used, density based methods such as Density Based Spectral Clustering for Applications with Noise (DBSCAN) are found to be better than other methods due to the noise inherently present in trajectory data [22]. Additional spectral clustering methods are also proposed [23], however it is shown that density based methods such as DBSCAN perform better [24]. For very large data sets, it is proposed to perform categorisation using hierarchical clustering to split the trajectories. This can be based on trajectory properties such as the origin or destination and it is proposed to subsequently apply DBSCAN to form clusters within these categories [25].

The research mentioned so far focuses on identifying air traffic flows. DBSCAN was also applied as clustering method for trajectory prediction on trajectory sets with a size of 5,000 to 8,500 trajectories [4], [9], [19], where clustering proved to be an effective method to reduce the size of training sets for the predictors. From both the research on air traffic flow identification and clustering before prediction it is concluded that DBSCAN is both the most commonly used and most promising method available and will therefore be used in this study.

However, it should be noted that the methods presented were developed to find air traffic flows and routes without any prior knowledge. Most airports around the world have Standard Arrival Routes (STARs) defined, which are publicly available and could form the basis of a clustering method as well. Therefore, it is chosen to compare DBSCAN, which is able to identify clusters by itself, to a clustering method based on STARs.

Finally, clustering based on spatial properties requires the distances between the trajectories. To determine these distances, several methods have been proposed. A common distance metric is the Hausdorff distance [26], which is a single metric based on the maximum distance between two trajectories. Also, Symmetrized Segment-Path Distance (SSPD) [27] is proposed, which determines the overall similarity between trajectories. It was found that Hausdorff is faster, whereas SSPD provides more accurate results [28], and it depends on the trajectory set which one is favourable. Therefore, both will be tested on the trajectory set.

C. Prediction using aircraft performance models

The current standard for trajectory prediction in ATM is systems using APMs, which are based on the Total Energy Model (TEM). The TEM sets the rate of work as performed

by the forces acting on the aircraft, for example the thrust forces applied by the engines, equal to the rate of potential and kinetic energy [29]. These models describing the aircraft's motion are combined with flight plan or intent data in order to predict the trajectory of an aircraft. Its inputs are the aircraft and engine properties, APPs.

The APPs provide information on each specific aircraft and are obtained from flight tests and specifications from the aircraft manufacturer. The most commonly accepted standard in this field is the Base of Aircraft Data (BADA) [2]. It comprises of over 160 unique sets of APPs which can be used to simulate aircraft performance.

Trajectories are predicted by simulating the aircraft over a filed flight plan. The benefit of this method is that the behaviour of the aircraft is according to the laws of physics. The disadvantage is that flight plan data is required in order to make a prediction while at the same time novel data types cannot be directly used for the predictions.

Capturing additional features in the input data to improve the predictions is indirectly possible by tailoring the APPs to a specific situation. The selection of APPs can be tailored by adapting the selection criteria such as the Calibrated Airspeed (CAS) or aircraft weight on specific use cases [30], [31], [32], [33], [34]. Also, it is proposed to develop new sets of APPs, tuned to specific situations [8], [35].

However, the focus of this study is on data-driven predictors without using APMs. Also, these methods have been compared to the current system by comparing it to the performance of BADA using various APM implementations and therefore they are not considered in this study. It is chosen to use BADA in an air traffic simulator to serve as a baseline, called the APM, as this is representative of the systems currently in place at ATC. Also, it provides the opportunity to compare this research to the work presented above in future research. Air traffic simulator 'BlueSky' [36] will be used as it is open source and capable of using the adapted and novel sets of APPs mentioned above.

D. Data-driven prediction using machine learning methods

Recently, efforts in the field of trajectory prediction have focused on predicting without using physical models. These methods use machine learning algorithms to capture the behaviour of aircraft without needing flight plans or aircraft specific information. Early research used simple neural networks to predict the vertical part of trajectories [37], to predict air traffic flow [38] and to predict delay [39]. All of these methods use historical trajectory data to train a predictor and predict by providing new trajectory points to this trained predictor. More detailed methods were developed to predict trajectories in the descent phase, comparing regression and decision tree methods such as Support Vector Machines versus Generalized Linear Models [12] and comparing Ridge Regression versus Gradient Boosting Machines and simple neural networks [40]. Also, it is proposed to apply functional regression on the trajectory prediction problem [6] However, it was concluded that this method is prone to overfitting, which is when an algorithm learns the training data so well that the performance on new data deteriorates, and a high sensitivity to noise and variation in the input parameters. For the climb phase, linear regression, simple neural networks and locally weighted linear regression are compared [5]. From these comparisons between regression, decision tree and shallow or simple neural networks, it is concluded that Gradient Boosting Machines, an ensemble method using decision trees, provides the best fit to trajectory prediction. Other methods are not found as suitable to capture the complex patterns of trajectory prediction and are therefore not considered in this work.

In contrary to simple neural networks, which comprise of only one to two layers of neurons, deep neural networks, comprising of more layers are better capable of capturing more complex relations in the data provided. These networks were tested on ETA prediction [11] and on trajectory prediction, where they were compared with Multiple-Linear Regression [4]. It was concluded that deep neural networks perform better on the problem at hand. However a comparison with other well performing methods such as gradient boosting has not been performed.

Observing that each trajectory essentially is a sequence of points in time and space, it can be represented by sets of time series. In order to use the relation between the current and previous points in a trajectory it is investigated whether Long Short-Term Memory networks could be used to perform trajectory prediction [3], [19]. These deep recurrent neural networks are designed for time series prediction and use an internal state to propagate information on previous time steps to the next one [41], which makes them suitable for the problem at hand [3], [19].

From this literature review, it is concluded that Gradient Boosting Machines (GBM), deep neural networks, also called 'Multilayer Perceptron' networks (MLP), and Long Short-Term Memory (LSTM) networks are the most promising data-based techniques to apply in this research. Also, no comparison between these three algorithms and an APM or other baseline simulation has been performed in literature.

E. Deep Neural Networks

Neural networks [42] consist of layers of neurons, also called cells, which perform computations on input to obtain a desired output. Each neuron represents a computational unit which has a bias and a weight, similar to coefficients in a regression function. The weighted input and bias of each neuron is summed and passed through an activation function such as a sigmoid or tanh. The activation function both regulates the output of that single neuron, as well as making it possible to predict non-linear behaviour. The equation for the output of single neuron x_i can be found in eq. (1),

$$x_i = f_{act}\left(\sum_{j=1}^{n} w_{i,j}x_j + b_i\right) \tag{1}$$

where x_j is the output of neuron j from the previous layer, $w_{i,j}$ is the weight of neuron i on the output of neuron j, b_i

the bias of neuron i and f_{act} the activation function. The output of a network can be determined by applying equation eq. (1) recursively. x_i is then defined as one of the output layer neurons of the model and by using eq. (1), each output of the previous layers x_j is determined. After each training run the error of the output is calculated using a loss function. Then, the weights of the network are updated accordingly to improve the prediction, using an optimiser. A neural network is deep if it consists of multiple layers, which enables a network to learn more complex relationships from the data.

F. Long Short-Term Memory Neural Networks

LSTMs are recurrent neural networks in which each cell uses both a new input as well as the output of the previous cell to provide a new estimate [41]. By propagating the output of each cell to the next one, a form of internal memory, the cell state, is present containing information on previous in- and outputs. This improves its performance when working with sequential or time series data with long-term time dependencies [41].

A schematic of a LSTM cell can be found in fig. 1, in which the horizontal arrow on the top represents the cell state C_t, which is received from the previous cell, adapted and then passed on to the next cell. Furthermore, several computational units are present in each LSTM cell, that work similar to the cells of a neural network and are represented by the yellow and red signs in fig. 1. Each part has its own weight w, bias b and predefined activation functions σ or $tanh$. They are defined as follows:

- **Forget gate**: Determines which information in cell state C_{t-1} should be kept and discarded based on previous output h_{t-1} and input x_t with a sigmoid function. It is represented by the σ on the left in fig. 1 and can be found in eq. (2).
- **Input gate**: Determines new candidate values for the cell state, $\tilde{C}_t$, by applying a $tanh$ function on h_{t-1} and input x_t, eq. (3). It also uses a sigmoid function to determine which part, i_t, of $\tilde{C}_t$ will be used to update C_t. See eq. (4). After this step, cell state C_t is updated using eq. (5). This gate is represented by the yellow σ and $tanh$ functions in the middle of fig. 1.
- **Output gate**: Consists of a sigmoid function that determines which information from h_{t-1} and x_t should be passed to the output, and can be found in eq. (6). A $tanh$ function determines the weight of cell state C_t which should be passed to the output, as can be seen in eq. (7).

$$f_t = \sigma\left(w_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2}$$

$$\tilde{C}_t = \tanh\left(w_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{3}$$

$$i_t = \sigma\left(w_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{4}$$

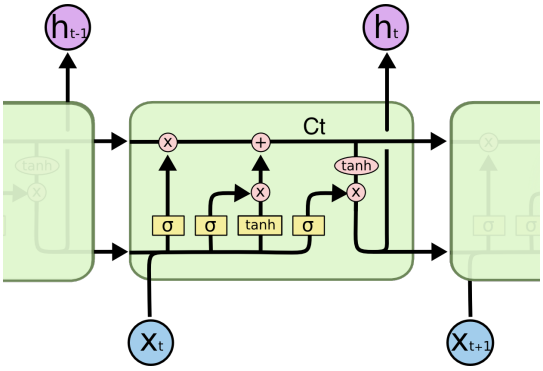


Figure 1. Schematic of a single LSTM cell, where cell state C_t is represented by the upper most horizontal black arrow, input x_t and output h_t is shown and the computational units are represented by the yellow and red boxes.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{5}$$

$$o_t = \sigma\left(w_o\left[h_{t-1}, x_t\right] + b_o\right) \tag{6}$$

$$h_t = o_t * \tanh\left(C_t\right)) \tag{7}$$

G. Gradient Boosting Machines

GBM [43] is an ensemble method which fits consecutive simple classification and regression trees [44] on training data, aiming to reduce the errors of the previous tree in each step. Each decision tree in a GBM is called a weak learner and is too shallow to capture the whole problem at once. It uses a loss function to determine the error of each weak learner and an additive model is used to add an additional tree after each iteration, while keeping existing trees intact. The algorithm performs well on classification and regression problems that rely on categorical indicators as it can efficiently use information from training examples with similar categories to find the correct outputs of the trees.

The algorithm is defined as follows [43]. The GBM, G_M, is initialised by fitting a constant function γ on the desired output y, and the optimal constant function is found using loss function L, with n being the number of values to be predicted:

$$G_0(x) = \gamma_{optimal} = min\Big(\sum_{i}^{n} L(y_i, \gamma)\Big) \tag{8}$$

then, a total of M weak learners h_m are added consecutively to this base function to find G_M. Each update of the model is called G_m and is determined using eq. (9):

$$G_m = G_{m-1}(x) + \gamma h_m(x) \tag{9}$$

where h_m is the weak learner added at step m and is determined by first computing the residual error of G_{m-1}, see eq. (9), and then training h_m on r_{im} instead of on y_i.

$$r_{im} = -\left.\frac{\partial L\left(y, G_{m-1}(x)\right)}{\partial G_{m-1}(x)}\right|_{x=x_i, y=y_i} \quad \text{for all } i = 1, \ldots, n \tag{10}$$

Finally, constant function γ in eq. (9) is optimised using eq. (11) and this procedure is repeated until M weak learners h_m have been fitted on the residuals and therefore G_M is constructed.

$$\gamma_{optimal} = min\Big(\sum_{i}^{n} L(y_i, G_m(x_i))\Big) \tag{11}$$

H. Density Based Spectral Clustering for Applications with Noise

DBSCAN [45] is an unsupervised clustering method and using two parameters it is able to find arbitrarily shaped clusters of various sizes within a single data set. These parameters are defined as follows:

- ϵ: or minimum distance. Defines how close two trajectories should be to each other in order to be called neighbours.
- MinPts: The number of neighbours a trajectory should have in order to be called a core trajectory.

Using these parameters, each trajectory in the data set is classified to construct clusters. A trajectory can be a core trajectory, which forms the basis of a cluster and has at least MinPts neighbours. The second category contains border trajectories, which are a neighbour to a core trajectory and are therefore part of a cluster, but do not have enough neighbours to be core trajectories. Finally, trajectories without sufficient neighbours and no core trajectories as neighbours are classified as noise trajectories, which are not a part of any cluster.

III. Methodology

The methodology presented here consists of two parts. First the preparation for trajectory prediction is presented. This includes data sourcing and processing, as well as trajectory clustering in order to speed up the predictions. Then, the three novel predictors are compared with the APM and assessed on their performance in both spatial and timing accuracy.

It is chosen to predict trajectories for aircraft arriving at Amsterdam Schiphol Airport (EHAM) as relevant data for this airport was available. Also, predicting arriving flights poses a greater challenge due to merging routes, vectoring and generally more ATC interference as opposed to departing aircraft.

A. Data sources and preparation

As shown in section II-A, ADS-B, aircraft weight and meteorological data are useful for trajectory prediction. However, aircraft weight data is not available and therefore only the aircraft type is used. Also, the influence of runway scheduling on trajectory prediction is investigated. Each step in the data preparation is shown in fig. 2 and explained in more detail in the following subsections.

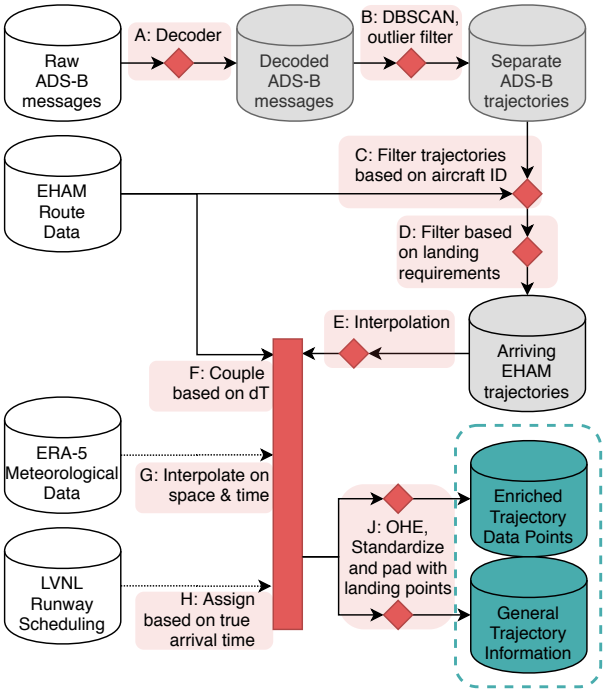


Figure 2. Data processing steps. The white sets on the left are the original sets from the data sources, the intermediate sets are grey and the resulting sets are indicated in teal. Red indicates an action on the data.

1) ADS-B data: The ADS-B data is sourced from a receiver located at TU Delft which has a coverage of over 400 km in radius [46]. The raw data first had to be decoded into numerical and categorical data entries, which is done using software developed by TU Delft [47], step A in fig. 2. Then, it is clustered into series of data which consist of entries from one trajectory, step B in fig. 2. The latter is done using DBSCAN [46], which clusters the data based on the similarity of the ADS-B messages and removes erroneous vertices by classifying them as noise points. Also, trajectories with errors are filtered using two additional methods. First of all, trajectories with a distance travelled of over 550 km are filtered out, considering that the range of the antenna is only 400 km and some erroneous trajectories were present in this range. Also, the aircraft speed is determined by calculating the distance travelled in each time step. Aircraft that are measured to fly faster than their maximum cruise speed are also removed from the data set. Flights arriving at EHAM are identified based on their ICAO aircraft ID and the location of the last three data points, which should be equal to the location of EHAM. These are steps C and D in fig. 2.

The resulting trajectories consist of irregularly spaced messages in the time domain due to noise and objects blocking the view of the antenna. To prevent the predictors from having to learn patterns in the timing, the trajectories are interpolated to a regular time step, step E in fig. 2. A time step of 4 seconds is chosen to reduce memory usage and computational effort in the prediction stage, while still allowing to use predictions for conflict detection. It is

assumed that the aircraft have a linear behaviour on this time scale and therefore the points are interpolated linearly.

2) Route data: Aircraft route data is used to identify aircraft flying to EHAM, step C in fig. 2, and to source additional parameters. No large open source data sets exist, however, all relevant information is published on the website of each airport daily in order to inform passengers. This information is obtained by means of web scraping. Amongst other parameters, the aircraft type is obtained. Additional features such as airline and origin are also gathered, however these features have not been proven to provide benefits for trajectory prediction and are therefore not used in this study. Also, 49% of the flights are executed by one airline, KLM, making it unrealistic to assess the influence of this parameter on the performance of a predictor. The origin is assumed to have little influence in this research as the direction of the origin is already taken into account when clustering by means of the spatial properties of the trajectories. The route data is coupled to the trajectories by verifying that the time of the last trajectory points matches with the executed arrival time, step F in fig. 2.

3) Meteorological data: The meteorological data used in this study is sourced from the 'ERA-5' data set [48]. It is available at a lat / lon grid of 0.25 x 0.25 degrees and at 37 altitudes ranging from ground level to $1 \cdot 10^5$ feet. Only the data up to 40,000 ft and only 3D wind vectors are used in this study.

The ADS-B data is enriched with the meteorological parameters by means of linear interpolation, step G in fig. 2. For each data point in the ADS-B set, the neighbouring latitude, longitude and altitude coordinates from the surrounding two time slots (in hours) of the ERA-5 set are taken, resulting in a weighted average of 2^4 values for each parameter.

4) Runway scheduling data: Runway scheduling at EHAM is dependent on wind conditions, but also on a set of agreements between the neighbouring towns and the government to minimise perceived noise. In view of the fact that the last part of the trajectory is dependent on the runway scheduling, it is investigated whether this parameter can be of use in the predictions.

The runway scheduling of EHAM is published on the website of the Dutch air traffic control (LVNL) in time intervals of 10 minutes for all runways. Similar to the route data, it is harvested from this website using webscraping. When predicting a trajectory, it is assumed that the runway scheduling at the scheduled time of landing is known. Using this assumption, the runway scheduling at this scheduled time of landing is added to the data set, step H in fig. 2, and it can be used for prediction.

5) Data preparation: An overview of the input parameters used for prediction can be found in table I. The machine learning based predictors require formatting of this data. Categorical data such as the aircraft type has to be converted to numerical data, which is done using One-Hot Encoding (OHE) [49]. OHE translates each unique value of a parameter to a column which can either be 0 or 1, false or true.

Table I
OVERVIEW OF THE PARAMETERS USED FOR TRAINING THE MACHINE LEARNING TRAJECTORY PREDICTORS, WHERE PARAMETERS WITHOUT UNIT REQUIRE ONE-HOT ENCODING.

ADS-B	**Additional Parameters**
lat/lon [deg]	3D wind vector [m/s]
altitude [ft]	runway scheduling [-]
ground speed [m/s]	aircraft type [-]
rate of climb [ft/min]	
heading [deg]	

Also, most machine learning algorithms expect input data with zero mean and unit variance. If the data is not scaled and standardised before training, the algorithms might overestimate the importance of parameters with a large absolute variance, such as altitude, and vice versa. Each parameter in the data set is scaled and standardised separately and μ and σ are based on the training set only for this procedure.

Finally, the neural networks are trained with sequences of an equal amount of time steps, which are trajectories of the same length. Common practice of sequence padding for machine learning is adding zeros, interpolation or extrapolation. Interpolation is not used, to keep the temporal component of the sequences. Both zero padding and extrapolation lead to unrealistic behaviour in the predictions, so it is chosen to pad all sequences to the same length with artificial 'landing' points at an altitude and with velocities of zero and the coordinate location of EHAM. These three steps are represented by step J in fig. 2.

B. Trajectory clustering

As detailed in section II-B, trajectories are clustered to increase the efficiency and accuracy of the prediction. Two methods are tested, DBSCAN and a STAR-based method, of which the methodology is provided in the following subsections. Both methods require distances to be calculated between trajectories, which will be presented in section III-C. All steps in this section can be found in the experiment diagram, fig. 3.

1) Clustering using DBSCAN: As explained in section II-H, DBSCAN requires tuning of ϵ and MinPts. This is done using a k-Nearest Neighbours (k-NN) algorithm and a k-distance plot in this study. The k-NN algorithm determines the number of neighbours a trajectory has for a range of distances. The ϵ should then be tuned to around the point of maximum curvature in the k-distance plot, which translates to the value where increasing the ϵ does not lead to a significant number of additional neighbours. The value for MinPts can then be

tuned to obtain the desired outcome by evaluating the results. The DBSCAN implementation of Python library Scikit-Learn [50] is used throughout this research.

2) Clustering based on STARs: All aircraft are assigned to a STAR in normal operations, which is defined in their flight plan. These flight plans, however, are not publicly available. Therefore, it is chosen to cluster the trajectories by calculating the distance of each trajectory to the known STARs. Trajectories are assigned to the STAR closest to the flown trajectory, assuming that the aircraft were also scheduled to use this STAR. The distance calculations for this method are done in 2D, considering that the STARs do not have overlap in altitude.

Ten out of the 14 STARs for EHAM are used, as can be seen in fig. 5b. HELEN 1A and PUTTY 1A are combined in one cluster as the trajectories have a lot of overlap due to shortcuts the aircraft are allowed to use in that area. Also, EELDE 1B, REKKEN 2B and NORKU 2B are not used for clustering as hardly any trajectories (< 0.1%) in the data set follow these routes.

C. Distance metrics

The trajectories are clustered based on their spatial properties, irrespective of time and where possible on their 3D spatial characteristics. As stated in section II-B, Hausdorff distances and Symmetrized Segment-Path Distance are compared in this study. Also, in order to speed up the distance calculations, the number of points per trajectory is reduced using the Ramer-Douglas-Peucker algorithm. The steps described in this section can be seen in fig. 3.

1) Hausdorff distance: The Hausdorff distance is obtained by calculating the distance between each point on trajectory A and its respective closest point on trajectory B. The maximum value of all distances obtained in this procedure is the Hausdorff distance. Its equation can be found in eq. (12), in which a and b are points on trajectories A and B respectively. However, this procedure has to be executed twice considering that $min_{b\in B}\{d(a,b)\}$ is not necessarily equal to $min_{b\in B}\{d(b,a)\}$. The maximum value of h(A,B) and h(B,A) is used as the final metric.

$$h(A,B) = max_{a\in A}\left\{min_{b\in B}\{d(a,b)\}\right\} \tag{12}$$

2) Symmetrized Segment-Path Distance: For SSPD [27], first the minimum distance between a point on trajectory A and each segment on B is determined using eq. (13),

$$D_{pt}(p_i^B, T^A) = min_{i_A\in[0,\dots,n_A-1]}\{D_{ps}(p_i^B, s_{i_A}^A)\} \tag{13}$$

where D_{pt} is the distance from point p on trajectory B to trajectory A and D_{ps} is the distance between point p on B and a segment s on trajectory A. Then, the average of these minimum distances of all points together forms the segment-path distance, D_{SPD} using equation eq. (14).

$$D_{SPD}(T^A, T^B) = \frac{1}{n_A}\sum_{i_A=1}^{n_A} D_{pt}(p_{i_A}^A, T^B) \tag{14}$$

Similar to the Hausdorff distance, this calculation has to be performed twice in order to make the result symmetric. In contrary to Hausdorff, the average result is used instead of the maximum result, using eq. (15).

$$D_{SSPD}(T^A, T^B) = \frac{D_{SPD}(T^A, T^B) + D_{SPD}(T^B, T^A)}{2} \tag{15}$$

SSPD provides a more global metric of the distance between two trajectories, in contrary to Hausdorff which provides a single maximum distance, not taking the rest of the difference between trajectories into account. SSPD was developed for comparing car trajectories [27] and its Python implementation is adapted in order to support 3D distance calculations for this specific use case on aircraft trajectories.

3) Ramer-Douglas-Peucker algorithm: To speed up the distance calculations, the number of points a trajectory consists of is reduced before applying SSPD or Hausdorff. This is done using the Ramer-Douglas-Peucker algorithm [51], which identifies the turning points of a trajectory. It keeps the points that provide the outline of a trajectory while disposing of points that are irrelevant for the overall pattern, as can be seen in Algorithm 1.

Algorithm 1 Ramer-Douglas-Peucker

Result: Simplified set of vertices $\{P_0, P_n\}$
1. Input: set of vertices $\{P_0, P_n\}$
2. Find vertex P_f with the largest distance to line $\overline{P_0P_n}$
if *distance* > ϵ **then**
  3. Apply algorithm recursively on set of vertices $\{P_0, P_f\}$
  4. Apply algorithm recursively on set of vertices $\{P_f, P_n\}$
else
  5. Remove P_f
end

where ϵ is the minimal distance or threshold parameter and is set at $1 \cdot 10^{-3}$ degrees, which translates to 2 nm or $3.7 \cdot 10^3$ meter at the latitude and longitude of the Netherlands. This procedure is only used to prepare the data for distance calculations and is not used for the prediction stage.

D. Aircraft Performance Models

As proposed in section II-C, the baseline simulations use aircraft performance models and parameters. The open-source air traffic simulator BlueSky [36] is used for the simulations in combination with the BADA 3.12 set of aircraft performance parameters [2]. The initial condition provided to BlueSky consists of the position, heading and CAS, extracted from the ADS-B data, and the aircraft type, extracted from the route data. The ADS-B ground speed is converted to CAS by first adding the wind vectors, essentially converting the ground

Table II
OVERVIEW OF THE INPUT PARAMETERS TO BLUESKY, OF WHICH THE INITIAL CONDITIONS ARE PROVIDED ONCE PER AIRCRAFT, THE VARIABLES FOR THE ROUTE ARE PROVIDED AS MANY TIMES AS THE AMOUNT OF WAYPOINTS IN THE ROUTE AND THE WINDFIELDS FOR THE ENTIRE RANGE OF THE ADS-B ANTENNA

Initial conditions	**Route**	**Windfields**
lat/lon [deg]	WPT [-]	lat/lon [deg]
alt [ft]	SPD [kts]	alt [ft]
heading [deg]	ALT [ft]	wind direction [deg]
CAS [kts]		wind magnitude [kts]

speed to TAS, as can be seen in eq. (16). The TAS is then converted to CAS using a conversion implemented in BlueSky [36].

$$\vec{v}_{TAS} = \vec{v}_{gs} - \vec{v}_{wind} \tag{16}$$

Apart from the initial conditions the route has to be provided, expressed in waypoints including speed and altitude commands. Considering that no flight plans are available, this information is deduced from the trajectories. All flights are headed for EHAM and use a STAR and instrument approach. These procedures include waypoints, altitudes and speed requirements and therefore provide all the information required for the simulation. Trajectories are assigned to a route using the same approach as described for the STAR-based clustering, assigning a trajectory to the route closest to the flown trajectory. However, for the baseline also the final approach is considered when assigning a route. Again, the number of vertices per trajectory is reduced using RDP and the distances between the trajectories and the STARs are calculated using Hausdorff and SSPD. These steps are visualised on the left in fig. 3.

On top of the previous, the windfields, as described in section III-A, are provided to BlueSky and are updated each hour. An overview of the inputs to BlueSky per aircraft and simulation is provided in table II. A calculation time step of 1 second is used and the location of the aircraft is logged. These logged locations form the predicted trajectory which can be evaluated. Aircraft are assumed to have landed when they have reached an altitude of less than 500 feet.

E. Machine learning predictors

After clustering the trajectories, each predictor is trained and tested on each cluster separately. As presented in section II, the predictors tested in this research are MLPs, LSTMs and GBMs and these are also shown in fig. 3. These all require selection and tuning of several parameters to optimise the results. It is noted that each cluster of trajectories requires its own tuning process in order to obtain the best possible result. However, considering that tuning each algorithm for each cluster requires a great amount of computational power, the predictors are tuned to perform well on one cluster and then applied in this configuration

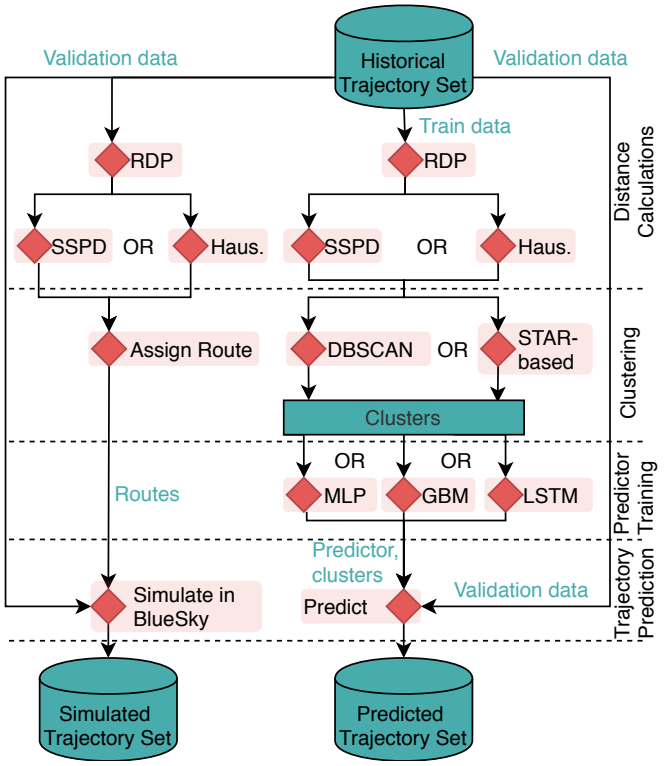


Figure 3. Experiment set-up. On the left, the steps of the APM are shown and on the right the novel predictors. Red indicates an action while teal indicates an in- or output.

on the other clusters as well. Considering that the type and complexity of the data is comparable over the clusters, it is assumed that this approach is sufficient in terms of parameter tuning. Also, the predictors are tuned and tested at a look-ahead time of 120 seconds, after which the best performing algorithms are also tested at larger look-ahead times.

1) Multilayer Perceptron: The MLP is trained by providing a sequence of points of the trajectory and the desired output, namely a new sequence of points which represents the predicted trajectory. The MLP is implemented using the Python based Keras library [52], in combination with machine learning platform Tensorflow [53]. Several parameters need to be chosen of which some can be derived from the nature of the problem. However, others need to be tuned in order to obtain the best result. The proposed model architecture can be found in table III.

Observing that trajectory prediction can be represented by a regression function, a Mean Squared Error (MSE) is used as loss function. The optimiser is the algorithm that updates the weights according to the loss to get the best prediction. Adaptive Moment estimation (ADAM) is used as it tunes the learning rate by itself and has low memory requirements. A $tanh$ activation function is used to enable the network to capture non-linear behaviour. Also, a dropout layer is used after each normal layer, which eliminates the 20 % least effect-

Table III
MODEL ARCHITECTURE OF THE MLP AND LSTM PREDICTORS.

Parameter	MLP	LSTM
Loss function	MSE	MSE
Optimizer	ADAM	ADAM
Activation function	$tanh$	$tanh$, σ
Dropout layer	20 %	20 %
Number of layers	3	1
Number of cells	200	300
Number of epochs	1000	500
Batch size	100	50

Table IV
MODEL ARCHITECTURE OF THE GBM PREDICTOR.

Parameter	GBM
Loss function	RMSE
Learning rate	0.01
Maximum tree depth	8
Number of weak learners	150
Column sampling	0.8
Time window	10

ive neurons during the training phase and thereby preventing overfitting. Finally, the number of layers, cells, epochs and the batch size were tuned by performing a grid search on the parameters. The batch size is the number of training samples that is provided to the network before updating the weights using the optimiser and the number of epochs is the amount of times the full set of training data is passed through the network.

2) Long short-term memory neural networks: Similar to the MLP, this predictor is implemented using Keras and Tensorflow and the same parameters need to be set. The proposed model architecture can be found in table III. The loss function and optimiser have been chosen with the same reasoning as for the MLP. The activation functions used in an LSTM have been specified in section II-F and again a dropout layer is used to reduce overfitting. Again the number of layers, cells, epochs and batch size have been tuned using a grid search. Each cell in an LSTM essentially is a small layer comprising of several computational units or cells, reducing the number of layers required to construct an optimal LSTM network in comparison with a MLP.

3) Gradient Boosting Machines: The XGBRegressor function of XGBoost [54], a distributed gradient boosting library with Python API, is used to construct the predictor. Again some parameters had to be set and tuned, which resulted in the model architecture as presented in table IV. The loss function used is similar to the loss function for the other predictors, using a Root Mean Squared Error (RMSE). The learning rate, maximum tree depth, number of weak learners, column sampling and batch size were tuned using a grid search of the most optimal parameters. A weak learner is a shallow decision tree, as introduced in section II-G, and its amount is tuned by checking for overfitting on the test set. The maximum tree depth is the maximum number of splits each weak learner can have. The column sampling is set at 80 %, meaning that each tree is fitted using a random sample of 80 % of the parameters, reducing overfitting. The time window is the number of previous points on the trajectory each training sample consists of. A prediction is obtained by providing a new sample of trajectory data to the GBM, which then returns a prediction value. It should be noted that XGBoost only provides univariate and single-step outputs. Therefore three separate models are trained for the latitude, longitude and altitude and each prediction only provides one point of the predicted trajectory, in comparison with the sequential predictions of the neural networks.

F. Evaluation of the predictors

Assessing the performance of a predictor is based on its accuracy in spatial terms, timing and the computational time it takes to train and predict. Spatial trajectory errors are the offset in location at a certain point in time, whereas timing errors are the offset in time at a specific point in space. In this study, the spatial error is split in the along track, cross track and vertical error. The timing error is measured at the point of landing, ETA prediction.

The along track error is determined using the difference between the distance travelled in the original direction of the track by the original and by the predicted trajectory. In this article, the along track error relative to the distance travelled is used. The cross track error is the perpendicular deviation from this original trajectory. For these distance calculations, the Earth is assumed to be flat, as the distances covered are small. The vertical error is calculated using the difference in altitude between the true and predicted trajectory.

An aircraft of a predicted trajectory is considered to have landed if the aircraft has reached EHAM, similar to the definition in section III-A and the speed of the aircraft is below the landing speed, for which values of 130 kts and 150 kts are used for ICAO Aircraft Approach Categories C & D respectively [55].

The results of the APM need to be adapted before evaluating, as one single simulation is performed instead of single predictions of 120 seconds ahead, resulting in an increasing look-ahead time and an increasing cumulative error. Therefore, at each time step the previous error is deducted, resulting in the error induced over the previous 120 seconds. It is noted that the cumulative error does influence the results of these simulations. However, the largest share of the lateral errors are due to the assigned route being different from the original trajectory flown and therefore these errors will in any case continue to persist. The cumulative vertical error does result in errors that persist in the prediction which otherwise wouldn't.

A different altitude will result in small differences in speeds and therefore distances travelled. However, observing that the deviation in altitude is low for a major part of the flight, and the remaining flight time after top of descent is small, this is assumed not to have a major effect on the results.

The computational time is measured during the training and the prediction phase. For both Keras as XGBoost the GPU implementation is used and all predictors are trained on a linux system using a NVIDIA GeForce GTX 1050 graphics card.

G. Validation

The results of this study are validated in two ways. First of all, the results of the newly developed predictors are compared with the APM or baseline simulations which are representative of currently used systems. Furthermore, the predictions are evaluated against the original trajectories. To ensure an unbiased evaluation, each predictor is trained using 2/3 of the data and tested using the remaining 1/3 of the data, as can be seen in fig. 3. Also, the ADAM optimizer has a random component and therefore it is chosen to perform these experiments four times and use the average of these predictions for evaluation.

IV. Results

A. Data preparation

The data is acquired in the period of 15-10-2019 to 20-01-2020 and this sourcing and preparation phase resulted in 39,418 historical trajectories combined with meteorological, route and runway scheduling data. The data is standardised, scaled and categorical values are One-Hot Encoded for the machine learning predictors. A clustered visualisation of the resulting trajectories is shown in fig. 5. 25.9% of the trajectories in the set are executed by a Boeing 737-800, and the rest of the trajectories is distributed over 40 other aircraft types. A more extensive analysis of the data is provided in Appendix A.

B. Trajectory clustering

Each clustering method is tested using both SSPD and Hausdorff distances. Prior to these distance calculations, applying the Ramer-Douglas-Peucker algorithm resulted in an average reduction of 93.65 % in the number of vertices in the trajectories, speeding up the distance calculations while keeping the outlines of the trajectories intact.

For the DBSCAN algorithms, first the correct parameters had to be found. The k-NN search in case of Hausdorff indicated that an ϵ of approximately 0.35 should be used and in the case of SSPD an ϵ of approximately 0.05. It should be noted that the distances are standardised and therefore do not represent a physical distance measure. It is chosen to test DBSCAN in combination with Hausdorff with an ϵ in the range of 0.25 to 0.40 with steps of 0.05 and similarly for SSPD a range from 0.02 to 0.08 with steps of 0.005 is used. For both methods, a MinPts value in a range of 5 to 120 in steps of 5 is tested. From these tests, the best results were identified. For this, a target of 5% outliers and ideal minimum cluster size of 1000 trajectories is used. Also, a visual inspection of the resulting clusters is performed. The resulting, most optimal, parameters can be found in table V.

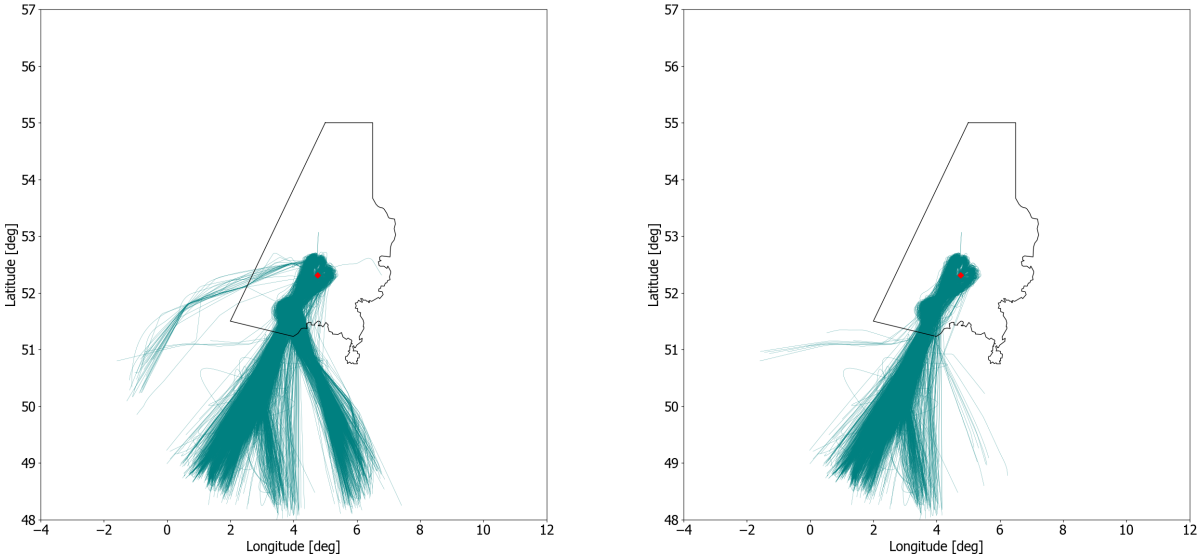
Table V
Overview of the clustering results using the 4 clustering methods. The computational time to perform the distance calculations and clustering is indexed at 100 = DBSCAN - Hausdorff.

	DBSCAN		**STAR**	
	Hausdorff	SSPD	Hausdorff	SSPD
ϵ/MinPts [-]	0.35 / 45	0.06 / 65	N/A	N/A
Number of clusters	6	7	10	10
Number of outliers	2940, 7.3%	4048, 10.1%	0	0
Comp. effort [-]	100	137.7	0.241	0.332

The difference in performance between SSPD and Hausdorff is visualised in the comparison of single clusters using the STAR based method, as can be found in fig. 4. It is observed that although the Hausdorff method is able to provide distances that result in clusters from roughly the same direction, it is not able to capture the details necessary in this trajectory set. In this case, a large share of the trajectories following HELEN are assigned to the DENUT cluster. SSPD is better capable of capturing these details as it provides a more average distance compared to Hausdorff. The same behaviour is present in most clusters for both DBSCAN and the STAR method. It should be noted that this accuracy comes at the cost of a larger computational effort, as stated in table V. The same assessment was performed when assigning the routes of the baseline simulations, resulting in the same conclusion. The results of assigning these routes are very similar to the results for clustering the trajectories and can be found in Appendix B.

The final results of clustering using DBSCAN and STARs in combination with SSPD can be found in fig. 5 and in table V. DBSCAN identifies a large share of the trajectories, 10%, as outliers, while not dividing the trajectories in the East and the South into separate clusters. Also, it identifies several fairly small clusters, which would result in less accurate results from the predictors in the next stage. Splitting the clusters in the East and the West requires a smaller ϵ, which would result in an even larger number of outliers. These would then be unavailable for training. On the other hand, to increase the size of the smaller clusters a larger ϵ or a smaller MinPts value is required, leading to even larger clusters in the East and South. When considering the West, it does perform well.

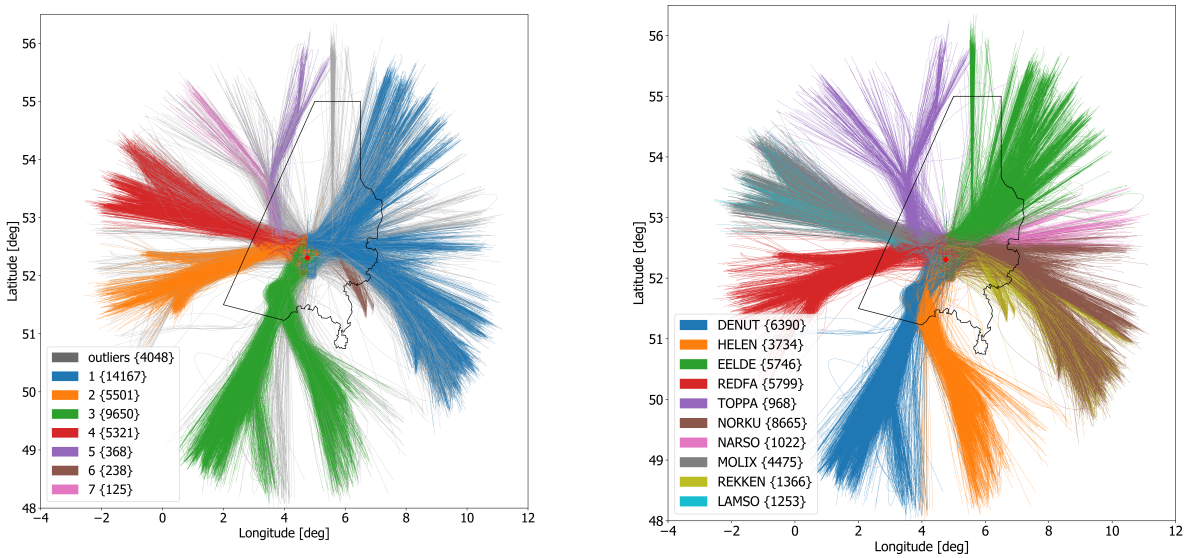
The STAR based method is able to separate clusters located close to each other and is able to find clusters of reasonable size with about 400 times less computational effort required. Also, it does not need to exclude outliers, such that all historical trajectories can be used to train predictors. Therefore, the STAR-based clustering method in combination with SSPD is used for the remainder of this study. An overview of



(a) Using Hausdorff distance metric

(b) Using SSPD distance metric

Figure 4. Comparison between the results of SSPD and Hausdorff distance metrics for the STAR-based clustering method. The resulting DENUT cluster is shown, the black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) DBSCAN clustering

(b) STAR based clustering

Figure 5. Result of clustering using SSPD as distance metric for both DBSCAN and the STAR based clustering approach. The black contour indicates the Dutch FIR and the red dot indicates EHAM.

the clustering results on a single-cluster level is provided in Appendix C.

C. Trajectory prediction

The evaluation proposed in section III-F is presented here. The results of the APM include all trajectories in the cluster set, where the LSTM, MLP and GBM results include the results for the test set, which is a random sample of 33%, only. Also, MLP and LSTM tests have been performed four times, as stated in section III, therefore the results include those four tests. These sets are all representative samples of the full problem. The predictors were tuned for the LAMSO cluster of the STAR-based clustering method as it is relatively small, speeding up the tuning process. The results for the experiments on this cluster are presented here. Additional results for the other clusters are shown in Appendix D.

1) Along track error: The relative along track error over the length of the flight per predictor on the LAMSO cluster can be found in the box plots in fig. 6. A positive along track error indicates that the aircraft is predict to fly too fast and

a negative along track error that the aircraft is behind on the original trajectory.

It is observed that the standard deviation of the APMs is in the same order of magnitude as the novel predictors when aircraft are far away from the airport, indicating similar performance in cruise. However, when approaching the airport larger differences in performance are observed. Furthermore, the MLP shows a large variance when the aircraft are further away from the airport and it does not perform as well as the LSTM and MLP predictors. The GBM predictor provides more constant predictions than the LSTM predictor, indicating that it performs better on the data at hand. In turn the LSTM predictor outperforms the MLP. Also, both the GBM as the LSTM predictors show an increase in variance in the last part, between 800 and 0 seconds to landing. This is approximately where the STAR ends and the aircraft fly to the final approach. However, also in this part the GBM predictor outperforms the others.

2) Cross track error: The cross track error for the experiments on the LAMSO cluster is shown in fig. 7, where no distinction is made between deviations to the left or the right of the original path.

A similar pattern as in the along track error is observed, with the APM having a larger mean cross track deviation and a larger variance than the novel predictors. Again, the GBM predictor outperforms the other predictors, while the MLP and LSTM predictors show similar performance. The cross track error also shows an increase in variance in the final part of the flight. Again, this is approximately where, in the Dutch FIR, the aircraft transition from the STAR to the final approach.

3) Vertical error: The vertical error is expressed in feet and can be found in fig. 8 for the LAMSO cluster. A positive vertical error indicates that the aircraft is predicted to fly at too large an altitude and a negative vertical error that the aircraft flying lower than the original trajectory. The APM provides good predictions of the altitude further away from the airport, in cruise, whereas the LSTM and MLP models have more difficulty in providing stable predictions in this flight phase. Interestingly, the novel predictors show only a small increase in variance when the aircraft approach the airport. The results of the APM show both a larger variance and a larger median, indicating that most of the predictions are too high. In the final phase however, the aircraft are simulated at too low an altitude.

Both the LSTM and MLP models show close to constant behaviour over the flights, with a reduction in deviation close to landing, indicating that the models perform well in the final approach. The MLP slightly outperforms the LSTM, whereas the GBM predictor outperforms both. It shows very accurate behaviour in cruise, while having similar performance during descent compared to the other predictors.

D. ETA error

The error in the predicted time of arrival for the LAMSO cluster can be found in fig. 9, where a positive ETA indicates that the prediction landed earlier than the original trajectory. It is observed that the APM simulations have a mean prediction error of zero, whereas the novel predictors have a slightly positive or negative offset. The variance in ETA prediction differs per predictor, although it is noted that the LSTM predictor performs worst and the GBM again shows the best performance. Overall, the variance in ETA error is larger than expected for a 120 seconds look-ahead time.

E. Computational time to train and predict

The time to train and predict for each predictor for the LAMSO cluster is shown in table VI. The training and prediction time for GBM includes the three separate models for each parameter (lat, lon, alt). It is observed that the LSTM model requires about 5 times more time to train, compared to the other models, whereas the GBM model is fast in training but slow in prediction. The latter is due to the fact that it predicts single values instead of multivariate sequential output, as stated in section III. The data in table VI is indexed as the exact values will differ significantly over data sets and hardware, but to provide an indication of the order of magnitude, the train time for the MLP model is equal to 860 seconds and the prediction time 2 seconds for the LAMSO cluster.

Table VI
COMPUTATIONAL TIME TO TRAIN AND PREDICT USING THE 3 NOVEL MODELS FOR THE LAMSO CLUSTER, INDEXED AT 100 = MLP

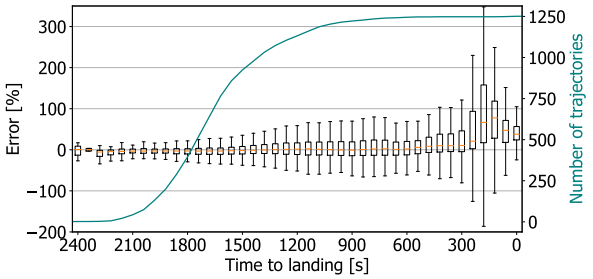
Predictor	Train time [index]	Prediction time [index]
MLP	100	100
GBM	87	1385
LSTM	505	109

F. Increased look-ahead time

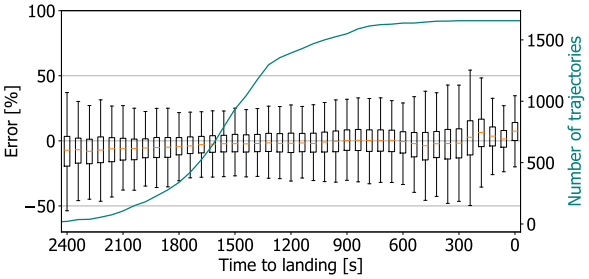
It is investigated how the performance of the predictors changes when the look-ahead time is increased. For this purpose, the along track error is analysed and only the two best performing predictors so far, LSTM and GBM, are tested. The result in terms of relative along track error of these tests are shown in fig. 10. It is observed that also at an increased look-ahead time, GBM is the best performing algorithm. Also, it is observed that although the performance deteriorates after increasing the look-ahead time over 2 minutes, due to the predictors being tuned for this value, they do not show worse performance when increasing the look-ahead time further. To provide an indication of the magnitude of the error for an increase in look-ahead time, the along track error is presented in fig. 11. It is observed that the magnitude of the along track error increases significantly in absolute terms, as expected.

G. Effect of runway scheduling data

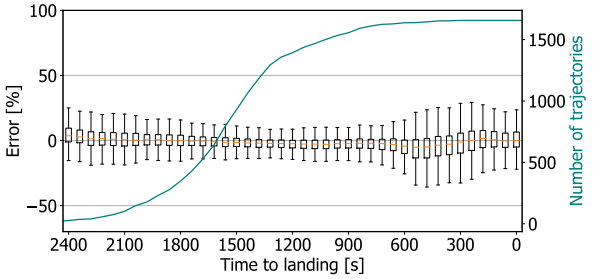
The runway scheduling data proposed in section III-A is used to train the 3 novel predictors. In fig. 12 it is observed that adding this data improves the cross track error in the third quarter of the flight. This is approximately the section



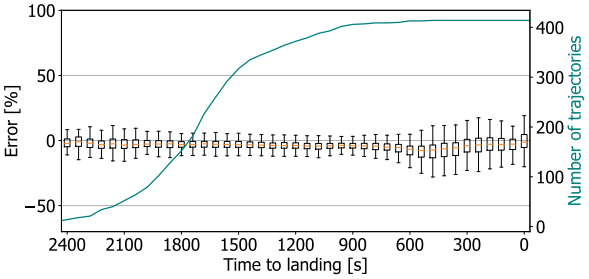
(a) APM, note the different scale of the y-axis compared to the other figures



(b) MLP

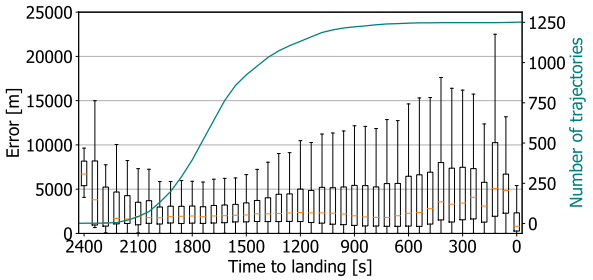


(c) LSTM

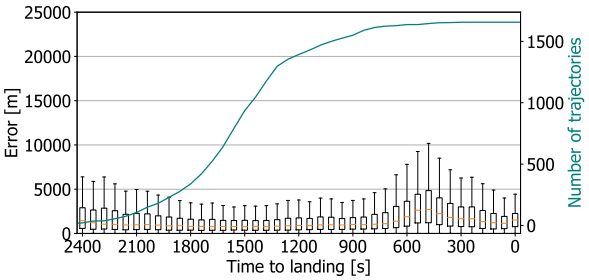


(d) GBM

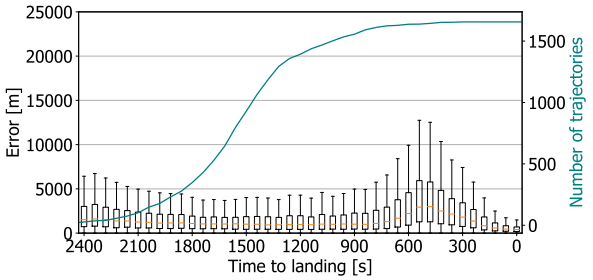
Figure 6. Relative along track error of the four prediction methods for the LAMSO cluster and a look-ahead time of 120 seconds. For a range of values for time to landing, boxplots are shown. The teal line indicates the number of trajectories taken into account in each box plot. The time to landing is measured from the moment of ADS-B interception by the antenna for each trajectory separately.



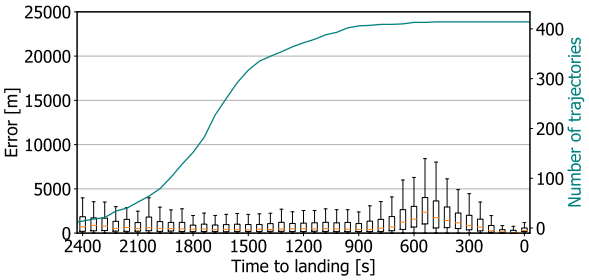
(a) APM



(b) MLP

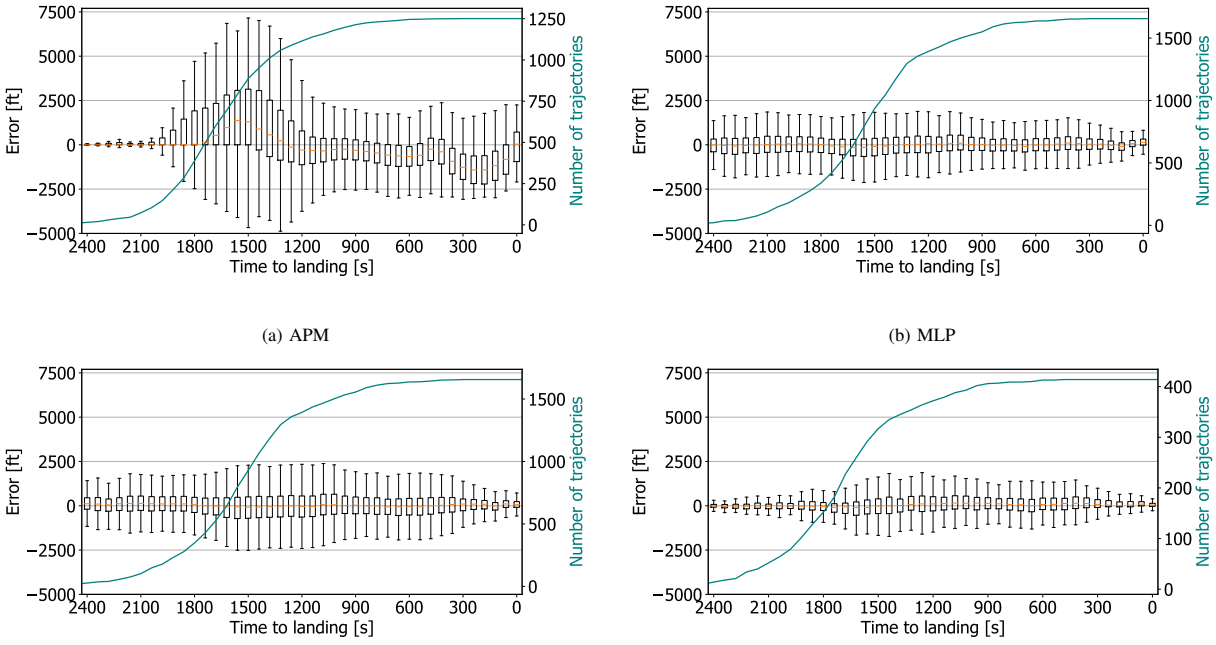


(c) LSTM



(d) GBM

Figure 7. Cross track error of the four prediction methods for the LAMSO cluster and a look-ahead time of 120 seconds. For a range of values for time to landing, boxplots are shown. The teal line indicates the number of trajectories taken into account in each box plot. The time to landing is measured from the moment of ADS-B interception by the antenna for each trajectory separately.



(a) APM

(b) MLP

(c) LSTM

(d) GBM

Figure 8. Vertical error of the four prediction methods for the LAMSO cluster and a look-ahead time of 120 seconds. For a range of values for time to landing, boxplots are shown. The teal line indicates the number of trajectories taken into account in each box plot. The time to landing is measured from the moment of ADS-B interception by the antenna for each trajectory separately.

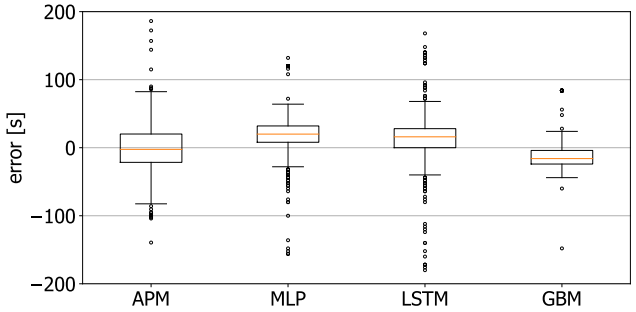


Figure 9. ETA error of the four methods for a look-ahead time of 120 seconds in the LAMSO cluster

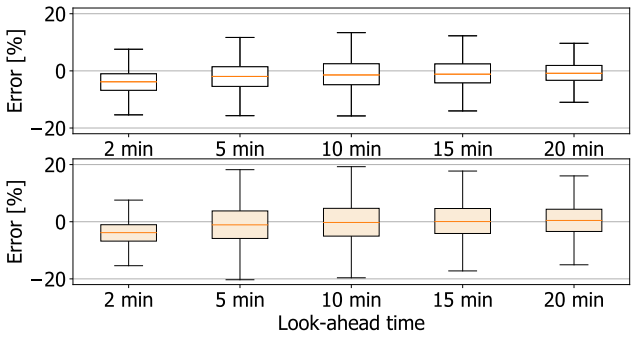


Figure 10. Relative along track error for an increasing look-ahead time of the GBM and LSTM predictors on the LAMSO cluster. The plot on the top represents the error for the GBM, whereas the bottom plot represents the relative along track error for the LSTM.

of the flight where the aircraft move from the STAR to their final approach. The improvements are small, with the mean error in the third quarter decreasing by 5.6 % and the standard deviation by 3.9 %. In the second and last quarter of the flight no significant change is observed. In the first quarter, which is the cruise phase, the performance of the predictor declined by 8.1 % in terms of mean error and by 6.0 % in terms of standard deviation. No significant difference is observed in the along track, vertical and ETA error.

V. Discussion

A. *Trajectory clustering*

It is observed that clustering using DBSCAN does not perform as expected. It identifies some very large clusters and some very small ones, while classifying a significant number of trajectories as outliers and it is computationally heavy. When tuning the parameters, is was not possible to find the right balance between the size of the clusters and the number of outliers. As this trajectory set includes an evenly spread, large number of trajectories from the East, a

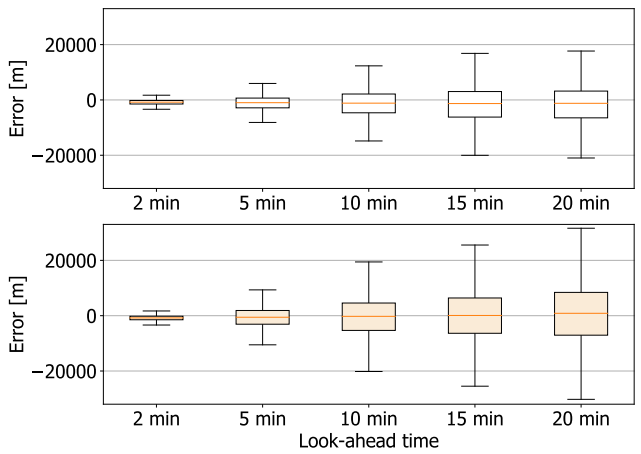


Figure 11. Along track error for an increasing look-ahead time of the GBM and LSTM predictors on the LAMSO cluster. The plot on the top represents the error for the GBM, whereas the bottom plot represents the relative along track error for the LSTM.

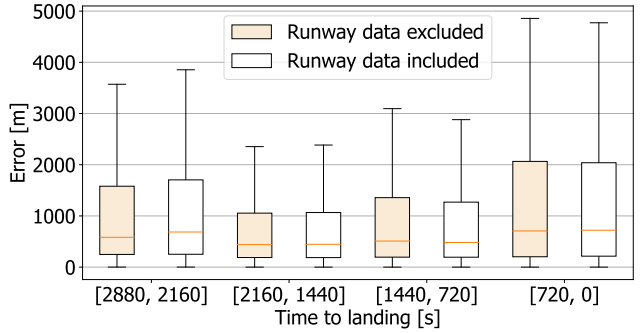


Figure 12. Cross track error for the LAMSO cluster, predicted using a GBM model with and without runway scheduling data in the training set. The data is split in four quarters measured in time to landing, the ranges indicate on which part of the flight the boxplot applies.

density based approach such as DBSCAN is not sufficient to capture differences between the sets of trajectories without classifying a large share of them as noise. In previous research, the DBSCAN method has proven to be useful to identify air traffic flows when no knowledge on the airspace is available. However, when applied as preparation for trajectory prediction where a minimum amount of trajectories should be classified as noise, it performs poorly. This is especially the case in an airspace with a scattered set of trajectories, such as the arriving aircraft to EHAM.

The STAR based clustering method does require knowledge on the airspace, however, it outperforms DBSCAN both in accuracy as in computational power required. A visual inspection of the results allows to see that it is able to distinguish clusters following routes which are close to each other without having to exclude a large share of the data set as outliers.

B. Trajectory prediction

The results show the same pattern over the three spatial metrics. The novel predictors outperform the APM which is as expected as these are capable of learning more complex patterns that cannot be captured in the simulations. The APMs perform reasonable from a timing perspective compared to the novel predictors.

The LSTM predictor was expected to outperform the other methods as it is known for its good performance on time series data. However, it performs better than the normal MLP only in terms of along track error and in ETA prediction, while it performs similar in cross track error and altitude prediction. The GBM model outperforms the LSTM on all metrics, which is not as expected. This indicates that the LSTM's capability of remembering past information to be used for predictions later on a sequence is not as valuable as expected for trajectory data. This, in combination with the significantly larger computational effort required to train LSTM predictors indicates that it is over-qualified for the problem at hand. The dependencies of the trajectory data over the long term are not so strong that it provides valuable information for prediction, making the main quality of LSTM networks redundant.

Whereas the focus of an LSTM network is on the previous points on a sequence, a GBM focuses more on information from other similar and relevant examples it has trained on. In this case, it is expected that the GBM performs better, especially in the later stages of the flight, as it is able to use information from other trajectories that had similar indicators. These indicators could then for example be the direction of the wind and runway scheduling, leading to the better performance in the segment from the STAR to the final approach, as is observed. Observing that these patters show resemblance of a classification problem within the regression, the GBM outperforms the neural networks.

Also, when increasing the look-ahead time of the predictions, the LSTM predictor did not perform better than the GBM predictor. However, it is observed that the relative error does not increase significantly, indicating that these techniques can be used for both short and medium term prediction. The effect of using runway scheduling data was not significant over the full length of the trajectory and is only found to be relevant on the point where the trajectories transit from their STAR to the final approach of the runways. However, as it is observed that this is the segment where the performance of all predictors deteriorates, it can be of use. Not only do the aircraft disperse from their STAR to the final approach in this stage, aircraft are also vectored off their normal routes by ATC to regulate the amount of aircraft on the final approach. To effectively account for the latter, additional research into the interaction of trajectory predictions should be performed.

The time to predict using GBMs is significantly larger as the latitude, longitude and altitude have to be predicted separately in the implementation used. However the absolute values are not of such magnitude that this poses a problem for implementation in real world applications. Finally, it is noted

that the LSTM model is found to be more resilient to parameter tuning, resulting in similar performance in a relatively wide range of settings. The MLP and especially the GBM models were harder to tune, in which the latter is found to be more prone to overfitting.

VI. Conclusions

The goal of this study was to identify the required preparation steps for data-driven trajectory prediction as well as identifying the best performing machine learning based trajectory prediction algorithm. In the first phase of this research it is concluded that using raw ADS-B data from an antenna in combination with various other open source data sets it is possible to construct a data set for trajectory prediction. The data sourcing and combining steps of the research have been set up in such a way that additional parameters can easily be implemented.

The preparation phase also includes trajectory clustering to increase efficiency and accuracy in the prediction phase. Two distance metrics used for clustering were tested and it is concluded that the Hausdorff distance is not capable of capturing the details required for this case of clustering and that the computationally heavier SSPD metric is to be used. It is concluded that clustering using DBSCAN does not meet all requirements for clustering prior to trajectory prediction. It classifies too many trajectories as outliers while not separating clusters from several major routes in the data set, which are located in the East and South of the Netherlands. A STAR-based clustering method, which essentially assigns each trajectory to a known STAR, is implemented instead. It outperforms DBSCAN in terms of accuracy, computational time required and in simplicity.

The second phase of this research encompasses trajectory prediction and predictors based on Multilayer Perceptrons, Long Short-Term Memory networks and Gradient Boosting Machines were tested. Also, a baseline simulation based on aircraft performance models is implemented for comparison. It is concluded that as expected, the three novel predictors outperform the baseline. However, where it was expected that the LSTM model would perform best, it is concluded that the LSTM and MLP models perform similarly and that Gradient Boosting Machines provides the best results on the problem at hand. This indicates that the LSTMs ability to use information from the distant past is not required for this case. This, in combination with the significantly longer time required to train the LSTM network, leads to the conclusion that LSTMs are over-qualified for trajectory prediction in terms of sequential prediction. From its relatively good performance in the flight phase where aircraft transition between their STAR and the final approach, it is concluded that the GBM predictor is more accurate as this phase has resemblance of a classification problem. The GBM focuses on any similar situation in the whole training set set rather than focusing on recent points in the sequence when predicting.

Finally, from the results obtained in this study it is concluded that Gradient Boosting Machines outperform the other novel predictors and the baseline overall, indicating that it is the best fit for trajectory prediction on arriving aircraft. However, further improvements should be investigated to increase the accuracy if these methods are to be implemented, especially in the later stages of the flight.

VII. Recommendations

Following this research, several recommendations for future research can be provided. First of all, in the data preparation phase it could be investigated whether additional parameters such as the origin and operator of a flight can be useful for prediction. Also, the accuracy in vertical terms could be improved by using the exact aircraft weight at each point on a trajectory. Furthermore, if a larger data set is used, it could be investigated to cluster the trajectories within a STAR-based cluster on their altitude, narrowing down the variance in the set even further.

Secondly, it is recommended to predict the speeds, both lateral as well as vertical, directly using the machine learning predictors. These speeds can already be deduced from the trajectory data for evaluation, but including it as output would provide additional features for the machine learning algorithms to optimise on. Note that these parameters are already included in this research, but only as input and not as output.

Also, in the prediction phase it is recommended to investigate a predictor which forecasts the pattern or route of a flight without adding the time component. Here, a conventional APM can be used to simulate the aircraft through this route. Doing so could lead to insights in the capabilities of both types of predictors, as a distinction can then be made between the patterns and the performance of aircraft. In this case, it is recommended to investigated a Gradient Boosting Machine to choose these routes.

Finally, collaborative trajectory prediction, in which predictions of multiple aircraft are determined at the same time, allowing for interaction between these predictions, should be investigated. Doing so could result an increase in accuracy in the later stages of the trajectories, where improvements are required.

References

[1] E. Fernández Calvo, J. M. Cordero, G. Vouros, N. Pelekis, T. Kravaris, G. Harris, G. Fuchs, N. Andrienko, G. Andrienko, E. Casado, D. Scarlatti, and P. Costas, "DART: A Machine-Learning Approach to Trajectory Prediction and Demand-Capacity Balancing," in *Seventh SESAR Innovation Days*, 2017.

[2] L. Sillard, F. Vergne, and B. Desart, "User Manual for the Base of Aircraft Data (BADA) Revision 3.12," *European Organisation for the Safety of Air Navigation*, vol. 12, no. 13, p. 93–129, 2014.

[3] Y. Pang, N. Xu, and Y. Liu, "Aircraft Trajectory Prediction using LSTM Neural Network with Embedded Convolutional Layer," in *Proceedings of the Annual Conference of the PHM Society*, 2019. [Online]. Available: https://doi.org/10.36001/phmconf.2019.v11i1.849

[4] Z. Wang, M. Liang, and D. Delahaye, "Short-term 4D Trajectory Prediction Using Machine Learning Methods," in *7th SESAR Innovation Days*, 2017.

[5] M. G. Hamed, D. Gianazza, M. Serrurier, and N. Durand, "Statistical prediction of aircraft trajectory: regression methods vs point-mass model," *10th USA/Europe ATM R&D Seminar*, 2013. [Online]. Available: https://hal-enac.archives-ouvertes.fr/hal-00911709

[6] K. Tastambekov, S. Puechmorel, D. Delahaye, and C. Rabut, "Aircraft trajectory forecasting using local functional regression in Sobolev space," *Transportation Research Part C*, vol. 39, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.trc.2013.11.013

[7] S. T. Barratt, M. J. Kochenderfer, and S. P. Boyd, "Learning Probabilistic Trajectory Models of Aircraft in Terminal Airspace From Position Data," *IEEE Transactions on Intelligent Transportation Systems*, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8551278/

[8] M. Hrastovec and F. Solina, "Machine learning model for aircraft performances," *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2014.

[9] Z. Wang, M. Liang, and D. Delahaye, "A hybrid machine learning model for short-term estimated time of arrival prediction in terminal manoeuvring area," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 280–294, 2018. [Online]. Available: https://doi.org/10.1016/j.trc.2018.07.019

[10] X. Guan, R. Lv, L. Sun, and Y. Liu, "A study of 4D trajectory prediction based on machine deep learning," *Proceedings of the 12th World Congress on Intelligent Control and Automation (WCICA)*, pp. 24–27, 2016.

[11] Z. Wang, M. Liang, and D. Delahaye, "Automated Data-Driven Prediction on Aircraft Estimated Time of Arrival," *Sesar Innovations Days*, 2018.

[12] A. de Leege, M. van Paassen, and M. Mulder, "A Machine Learning Approach to Trajectory Prediction," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Delft, 2013.

[13] J. Sun, *The 1090MHz Riddle*. TU Delft. [Online]. Available: https://mode-s.org/decode/adsb/introduction.html

[14] J. Zhang, W. Liu, and Y. Zhu, "Study of ADS-B data evaluation," *Chinese Journal of Aeronautics*, vol. 24, no. 4, pp. 461–466, 2011.

[15] S. Mondoloni and I. Bayraktutar, "Impact of factors, conditions and metrics on trajectory prediction accuracy," in *Proceedings of the 6th USA/Europe Air Traffic Management Research and Development Seminar, ATM*, 2005, pp. 305–314.

[16] S. Mondoloni, M. Paglione, and S. M. Green, "Trajectory modelling accuracy for air traffic management decision support tools," *International Congress of Aeronautical Sciences (ICAS)*, pp. 1–10, 2002.

[17] J. Rudnyk, J. Ellerbroek, and J. Hoekstra, "Trajectory Prediction Sensitivity Analysis Using Monte Carlo Simulations," in *Aviation Technology, Integration, and Operations Conference*, 2018.

[18] P. Weitz, "Determination and visualization of uncertainties in 4D-trajectory prediction," *Integrated Communications, Navigation and Surveillance Conference, ICNS*, pp. 1–9, 2013.

[19] Y. Liu and M. Hansen, "Predicting Aircraft Trajectories: A Deep Generative Convolutional Recurrent Neural Networks Approach," Tech. Rep., 2018. [Online]. Available: https://arxiv.org/pdf/1812.11670v1.pdf

[20] C. Jesse, H. Liu, E. Smart, and D. Brown, "Analysing Flight Data Using Clustering Methods," in *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2008, pp. 733–740. [Online]. Available: https://link.springer.com/content/pdf/10.1007/978-3-540-85563-7_92.pdf

[21] K. Leiden and S. Atkins, "Trajectory Clustering for Metroplex Operations," in *11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2011.

[22] M. Gariel, A. N. Srivastava, and E. Feron, "Trajectory clustering and an application to airspace monitoring," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1511–1524, 2011.

[23] M. Enriquez, "Identifying Temporally Persistent Flows in the Terminal Airspace via Spectral Clustering," in *Air Traffic Management R&D Seminar*, 2013. [Online]. Available: http://atmseminar.org/seminarContent/seminar10/papers/183-Enriquez_0124131201-Final-Paper-4-12-13.pdf

[24] M. Condé, R. Murça, R. J. Hansman, H. Balakrishnan, R. Delaura, R. Jordan, and T. Reynolds, "Trajectory Clustering and Classification for Characterization of Air Traffic Flows," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016. [Online]. Available: http://arc.aiaa.org/doi/10.2514/6.2016-3760

[25] E. Salaün, M. Gariel, A. E. Vela, and E. Feron, "Aircraft proximity maps based on data-driven flow modeling," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 2, pp. 563–577, 2012.

[26] H. Blumberg, "Hausdorff's Grundzüge der Mengenlehre," *Bulletin of the American Mathematical Society*, vol. 33, no. 6, pp. 116–129, 1920.

[27] P. Besse, B. Guillouet, J.-M. Loubes, and R. François, "Review and Perspective for Distance Based Trajectory Clustering," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 11, pp. 3306–3317, 2016.

[28] L. Basora, J. Morio, and C. Mailhot, "A trajectory clustering framework to analyse air traffic flows," in *SESAR Innovation Days*, 2017.

[29] F. D. B. Sillard, L.; Vergne, "Eurocontrol Synonym Aircraft Report for the Base of Aircraft Data (BADA) - Revision 3.7," Tech. Rep., 2009.

[30] R. Alligier, D. Gianazza, and N. Durand, "Learning the aircraft mass and thrust to improve the ground-based trajectory prediction of climbing flights," *Transportation Research Part C: Emerging Technologies*, vol. 36, pp. 45–60, 2013. [Online]. Available: http://dx.doi.org/10.1016/j.trc.2013.08.006

[31] R. Alligier and D. Gianazza, "Machine Learning and Mass Estimation Methods for Ground-Based Aircraft Climb Prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3138–3149, 2015. [Online]. Available: https://hal-enac.archives-ouvertes.fr/hal-01181173

[32] R. Alligier, D. Gianazza, and N. Durand, "Machine Learning Applied to Airspeed Prediction During Climb," in *11th USA/EUROPE Air Traffic Management R&D Seminar*, 2015.

[33] M. Hrastovec and F. Solina, "Prediction of aircraft performances based on data collected by air traffic control centers," *Transportation Research Part C: Emerging Technologies*, vol. 73, pp. 167–182, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.trc.2016.10.018

[34] A. Hadjaz, G. Marceau, P. Savéant, and M. Schoenauer, "Online Learning for Ground Trajectory Prediction," in *SESAR 2nd Innovation Days*, 2012.

[35] M. Hrastovec, "Prediction of aircraft trajectories for air traffic control using machine learning approaches," Ph.D. dissertation, 2018.

[36] J. M. Hoekstra and J. Ellerbroek, "BlueSky ATC simulator project: an open-data and open-source approach," *Proceedings of the 7th International Conference on Research in Air Transportation*, 2016.

[37] Y. Le Fablec and J. M. Alliot, "Using Neural Networks to predict aircraft trajectories," in *Proceedings of the International Conference on Artificial Intelligence*, 1999.

[38] T. Cheng, D. Cui, and P. Cheng, "Data mining for air traffic flow forecasting: A hybrid model of neural network and statistical analysis," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, vol. 1, pp. 211–215, 2003.

[39] N. Takeichi, R. Kaida, A. Shimomura, and T. Yamauchi, "Prediction of Delay due to Air Traffic Control by Machine Learning," in *AIAA Modeling and Simulation Technologies Conference*, 2017.

[40] R. Alligier, D. Gianazza, and N. Durand, "Predicting Aircraft Descent Length with Machine Learning," in *7th International Conference on Research in Air Transportation (ICRAT)*, 2016. [Online]. Available: https://hal-enac.archives-ouvertes.fr/hal-01353960

[41] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] J. Hertz, A. Krogh, and R. Palmer, "Introduction To The Theory Of Neural Computation," 1991.

[43] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[44] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification And Regression Trees*. CHAPMAN & HALL/CRC, 1984.

[45] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, vol. 96, no. 34, 1996.

[46] J. Sun, J. Ellerbroek, and J. Hoekstra, "Large-Scale Flight Phase Identification from ADS-B Data Using Machine Learning Methods," in *7th International Conference on Research in Air Transportation*, 2016.

[47] J. Sun, "SIL: Python Client for Mode-s Beast Raw Data," 2019. [Online]. Available: https://github.com/junzis/sil

[48] "ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate," 2017. [Online]. Available: https://cds.climate.copernicus.eu/cdsapp#!/home

[49] S. L. Harris and D. M. Harris, "Sequential Logic Design," in *Digital Design and Computer Architecture*. Morgan Kaufmann, 2016, pp. 108–171. [Online]. Available: https://doi.org/10.1016/B978-0-12-800056-4.00003-0

[50] F. Pedregosa, R. Weiss, and M. Brucher, "Scikit-learn : Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[51] D. H. Douglas and T. K. Peucker, "Algorithms for the Reduction of the Number of Points Required To Represent a Digitized Line or Its

Caricature," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.
[52] F. Chollet, "Keras," 2015. [Online]. Available: https://github.com/fchollet/keras
[53] M. Abadi, A. Agarwal, and P. Barham, "Tensorflow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," 2015. [Online]. Available: https://www.tensorflow.org/
[54] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. 13-17, pp. 785–794, 2016.
[55] "ICAO - PANS-OPS, (Doc 8168), Procedures for Air Navigation Services - Aircraft Operations," Tech. Rep., 2006.

II

Scientific Paper Appendices

A

Route data analysis

In this appendix an analysis of the data gathered between 16-10-2019 and 20-01-2020 is presented. A similar overview is provided in the Preliminary Report, part III, however, additional outlier detection has been performed for the final result as presented in part I. Seeing as the aircraft type has been shown to be of importance for the final results, an update of the data analysis is provided here. Next to the aircraft type, the airline, origin and scheduled time of arrival are also analysed. These are not used for the final results of this study, however, it is recommended to investigate their effect on trajectory prediction in further research. All information in this appendix originates from the route data.

First, an overview of the full data set will be provided in section A.1, after which the contents of the LAMSO cluster only will be shown in section A.2 as the main experiments have been performed on this cluster.

A.1 Full trajectory set

The aircraft types of the trajectories in the set are shown in fig. A.1. It is observed that at 25.9% of the total amount of trajectories, the Boeing 737-800 is the most common aircraft type in the data set. The rest of the aircraft are distributed over another 40 aircraft types. In fig. A.2 it can be seen that 49% of the aircraft belong to KLM, while the remainder of the trajectories are from aircraft belonging to a total of 82 other airlines.

The origin of the trajectories is distributed over in total 279 origins present in the full set, of which the more frequent origins are shown in fig. A.3. Finally, in fig. A.4 it is observed that the majority of the trajectories are scheduled to arrive during day time with a peak in arrival from 7.00 to 9.00 o'clock and between 18.00 and 20.00 o'clock.

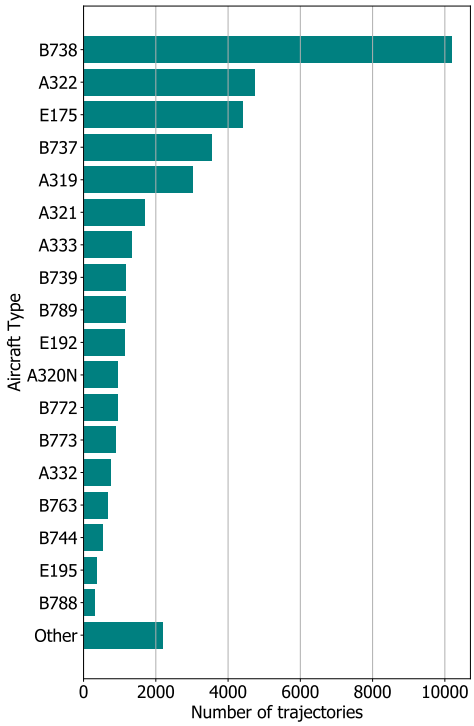


Figure A.1: Frequency of occurrence of 41 aircraft types in the full trajectory set.

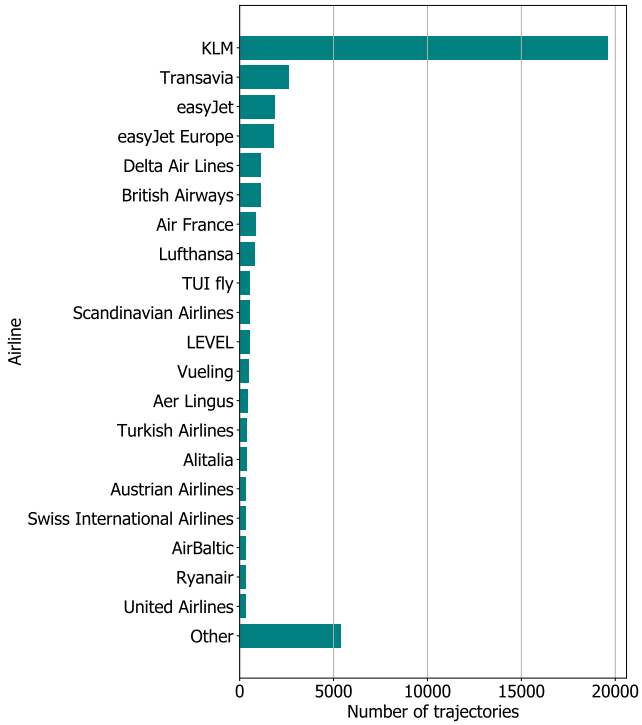


Figure A.2: Number of trajectories belonging to each airline present in the full trajectory set.

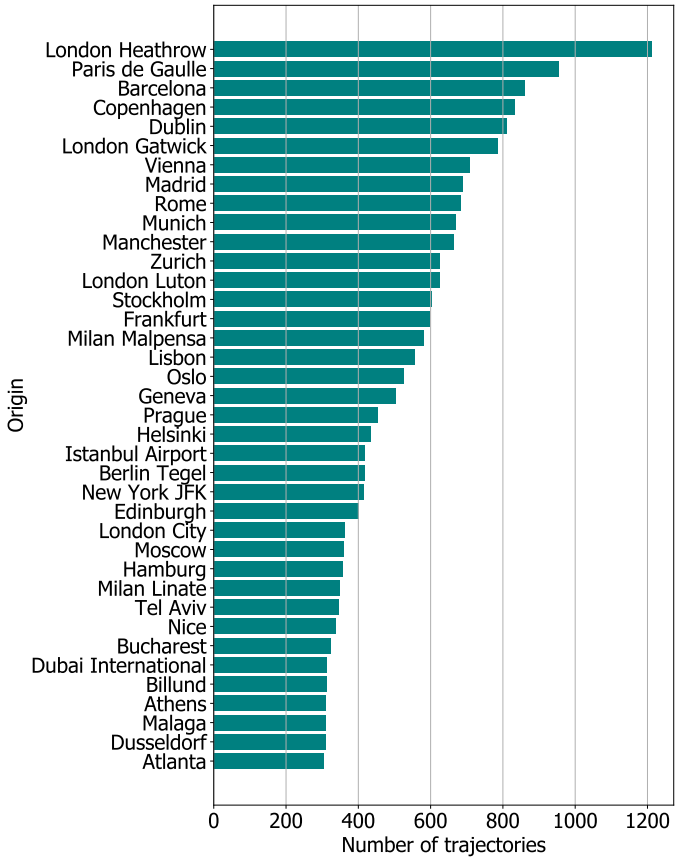


Figure A.3: Origin of the trajectories in the full trajectory set, excluding 19,797 trajectories which originate from other, less frequently occurring departure airports.

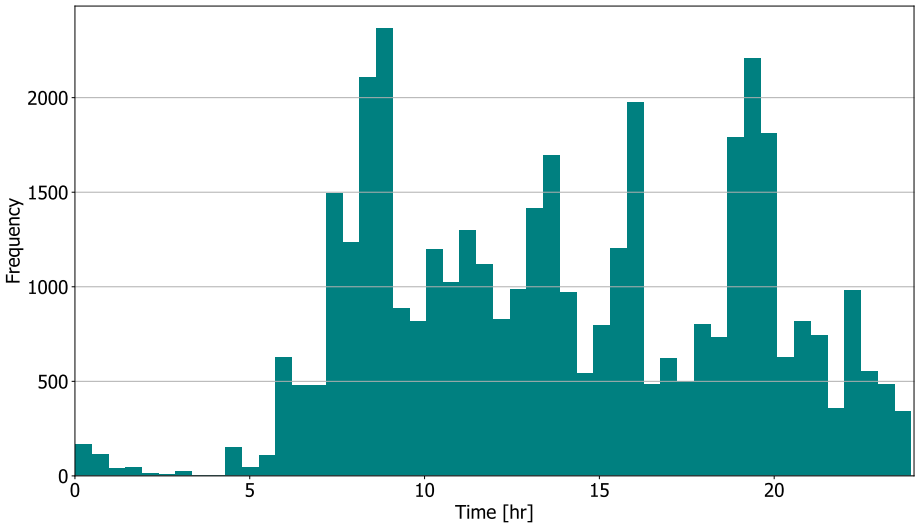


Figure A.4: Distribution of the scheduled arrival time of the trajectories in the full trajectory set.

A.2 LAMSO cluster

The LAMSO cluster comprises of 1,253 trajectories originating from the West. As shown in fig. A.5, it has a similar distribution of trajectories over the aircraft types compared to the full set, although the Boeing 737-800 is less prevalent in this cluster. The airlines in this cluster are mainly UK and USA-based airlines, which is as expected for this STAR. Its distribution is shown in fig. A.6.

Similar to the airlines, the origins of the trajectories in this set are mainly in the UK and USA, as is observed in fig. A.7. Finally, the distribution of scheduled time of arrival over the day is similar to the distribution of the full data set, apart from one large peak in arrivals at 16.00 o'clock, fig. A.8.

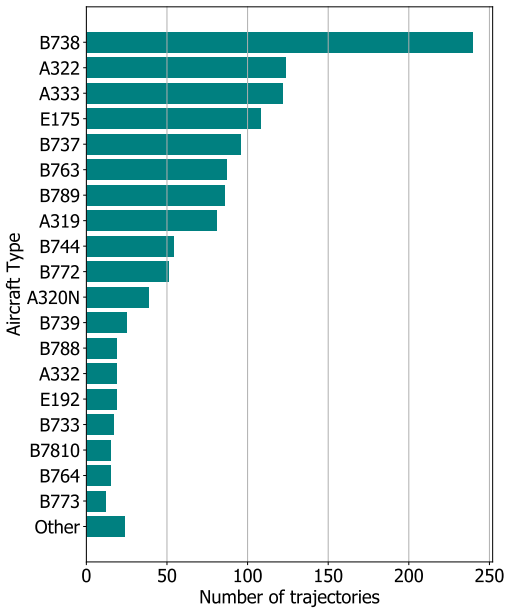


Figure A.5: Frequency of occurrence of 41 aircraft types in the LAMSO cluster.

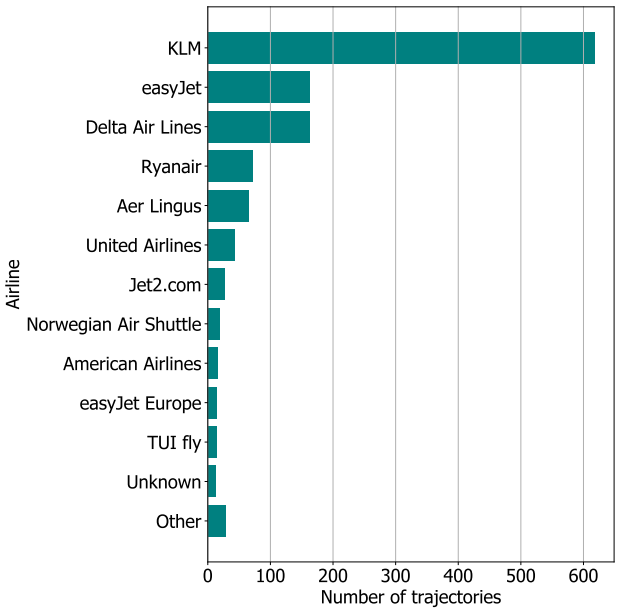


Figure A.6: Number of trajectories belonging to each airline present in the LAMSO cluster.

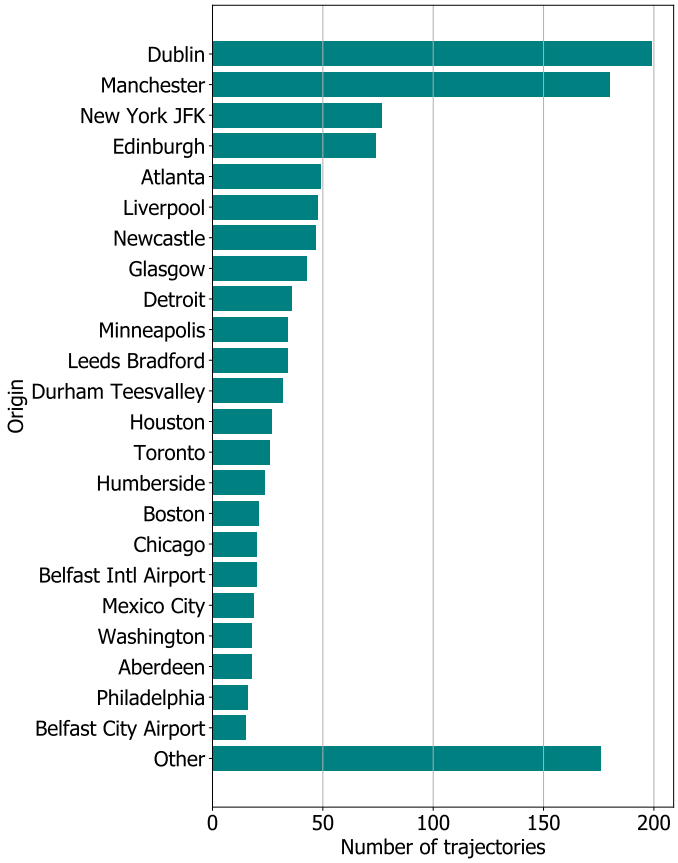


Figure A.7: Origin of the trajectories in the LAMSO cluster.

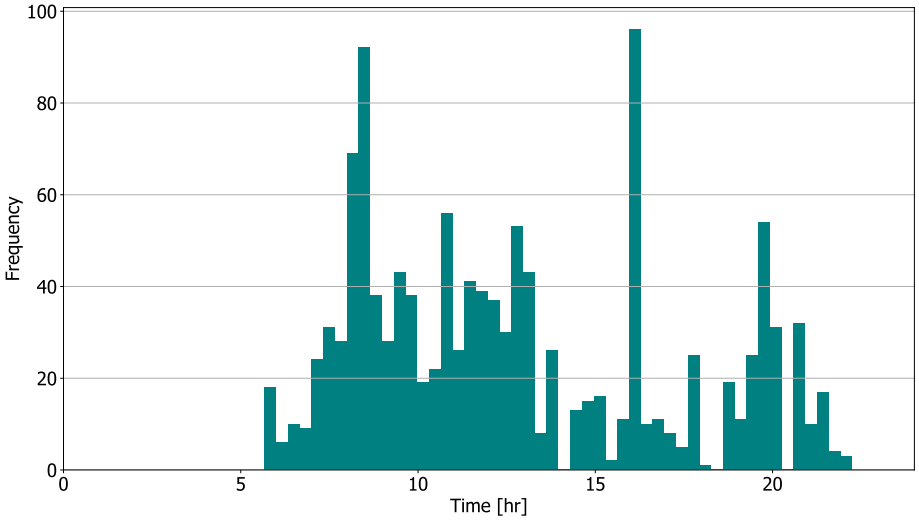


Figure A.8: Distribution of the scheduled arrival time of the trajectories in the LAMSO cluster.

B

Route assignment for the baseline simulations

As explained in the article, part I, the baseline simulations require a route or flight plan as input. To provide an insight in the quality of these routes four examples of trajectories with their assigned routes are shown and a qualitative analysis is performed. Each of these examples represents a pattern observed in a part of the assigned routes. A cause for these patterns and an explanation of why the behaviour is accepted is provided for each case. The four cases identified are as follows:

1. Direct-to's and vectoring, small deviations from the routes
2. Following undefined or non-official routes
3. Transition from STAR to final approach is not clearly defined
4. Skipping waypoints on a regular basis

It should be noted that examples with relatively little data are chosen to be able to visually see these patterns in the trajectory plots. In total 144 combinations of STARs and final approaches were used to construct the routes for the baseline simulations.

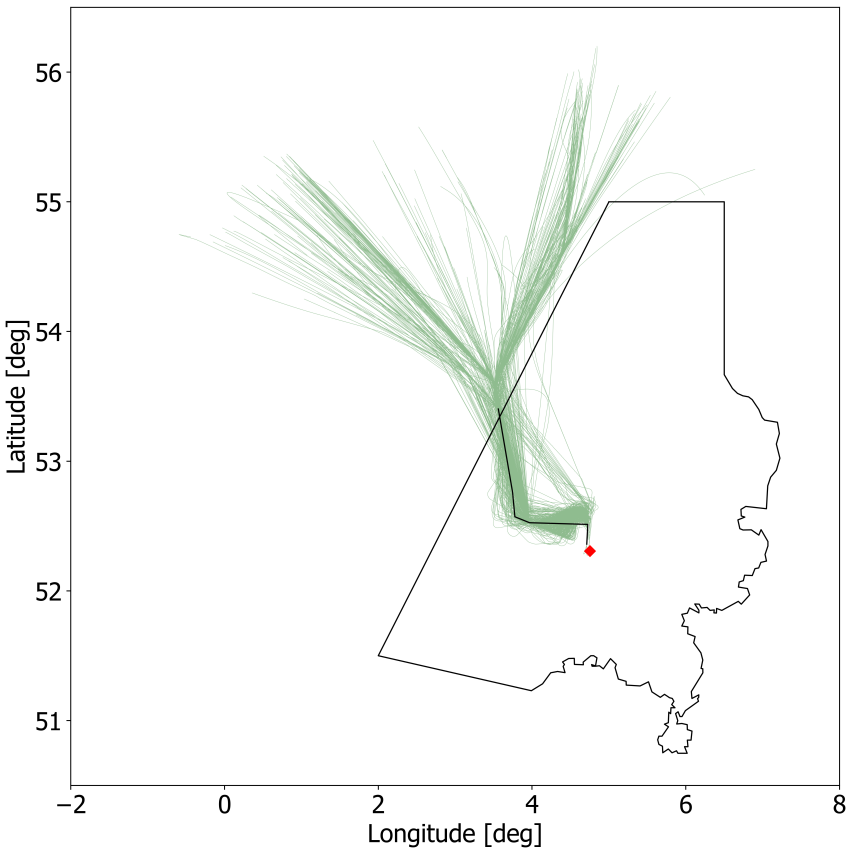
A clear example of vectoring and direct-to's is observed in the trajectories following the TOPPA STAR to runway 18R fig. B.1a. Some aircraft are allowed to fly from the border of the FIR directly to the final approach, skipping several waypoints. Also, a significantly larger number of aircraft are vectored off the standard route prior to the last turn before entering the final approach. Seeing as this behaviour would not have been included in any predefined route, it is concluded that this is the maximum level of accuracy that can be obtained using the information available.

In the second example, fig. B.1b, the trajectories assigned to runway 36R are shown. Here, it is observed that some aircraft fly via the East of Schiphol airport instead of via the route depicted in the figure. No standard route is available representative of the route flown. Therefore, the route it is assigned to is the best fit for these trajectories, although large differences are present.

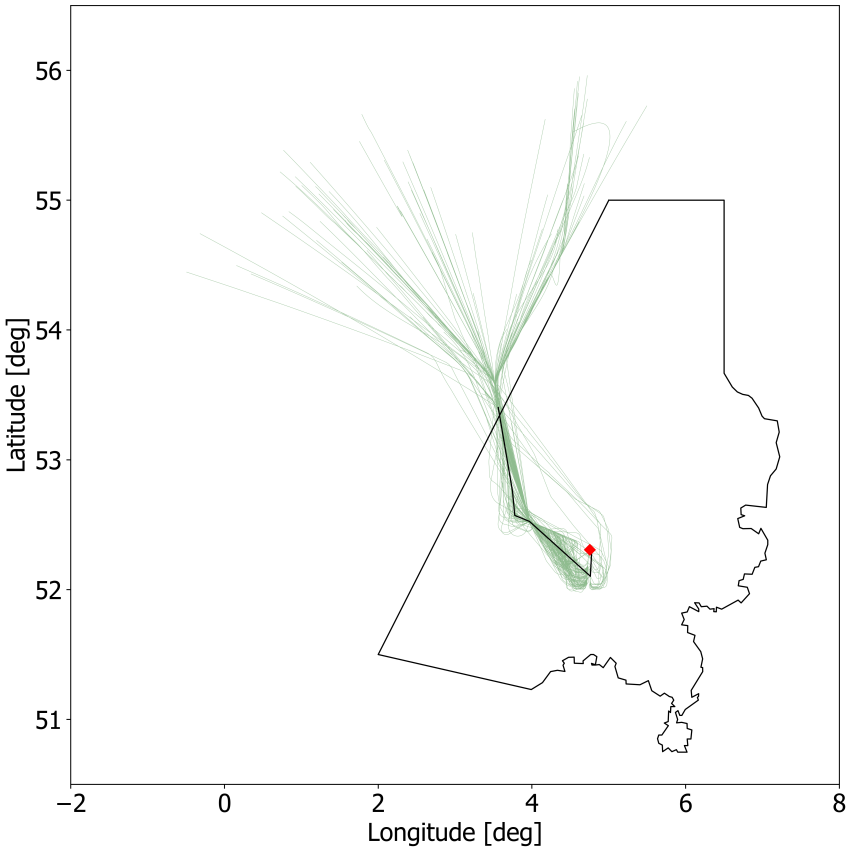
The third case shows how the transition from the STAR to the final approach is under defined. In fig. B.2a the trajectories following PUTTY towards runway 18R are shown. It is observed that a variety of options to transition from the STAR to the final approach is used, some of which fly via the west of EHAM while others fly over EHAM to approach runway 18R from the northeast. However, seeing as these patterns are not defined in official routes this behaviour is accepted.

The fourth pattern, skipping waypoints on a regular basis, is in the basis the same as direct-to's granted to aircraft. However, in both fig. B.2a and fig. B.2b it is observed that nearly all aircraft skip one waypoint, HSD. To improve the accuracy of the baseline simulations, it could be investigated to adapt the standard routes to the most common trajectories followed by aircraft, however in this study only the official routes are used.

Concluding, by means of visual inspection it is observed the approach of assigning routes based on the minimum distance between a trajectory and the standard routes provides acceptable results. Improvements could be investigated to make these routes more accurate, however this level of accuracy is assumed to be representative of how aircraft are routed in real life.

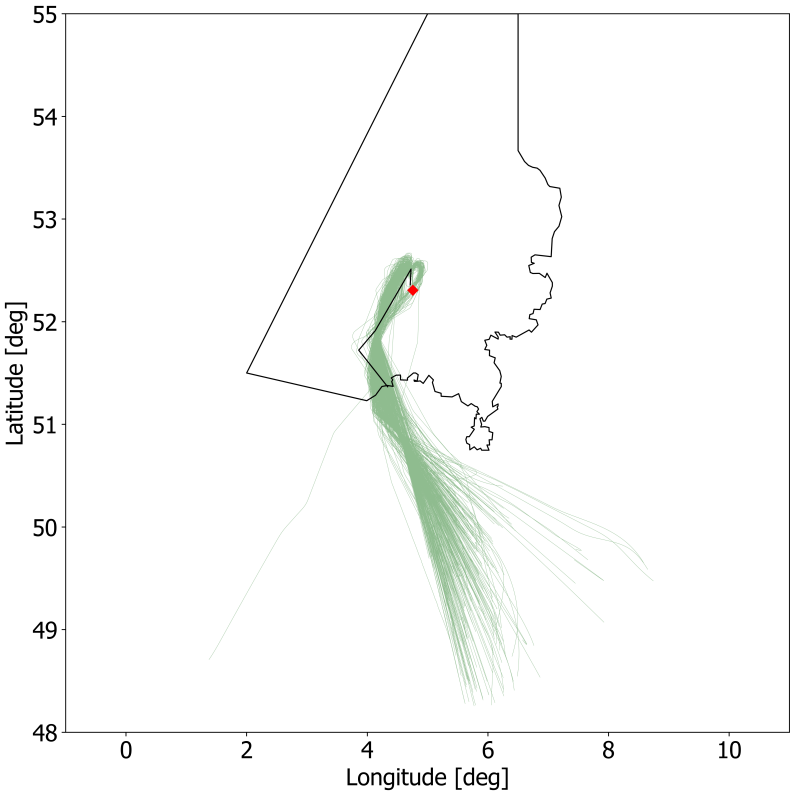


(a) TOPPA - 18R, containing 492 trajectories.

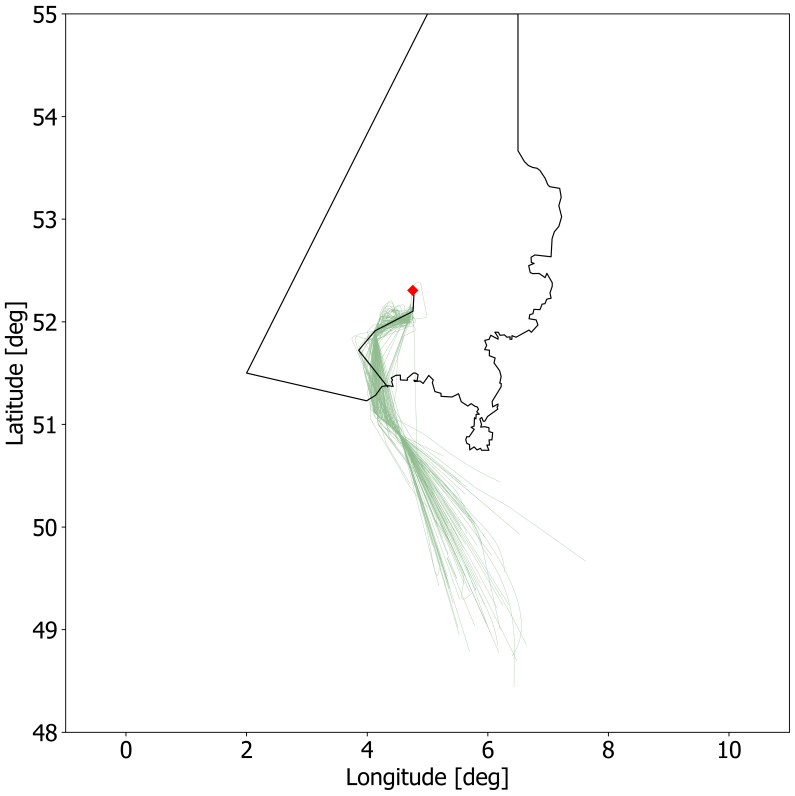


(b) TOPPA - 36R, containing 126 trajectories.

Figure B.1: Trajectories assigned to runways 18R and 36R following TOPPA. The black line indicates the route, the red dot indicates EHAM and the black contour the Dutch FIR.



(a) PUTTY - 18R, containing 506 trajectories.



(b) PUTTY - 36R, containing 106 trajectories.

Figure B.2: Trajectories assigned to runways 18R and 36R following PUTTY. The black line indicates the route, the red dot indicates EHAM and the black contour the Dutch FIR.

C

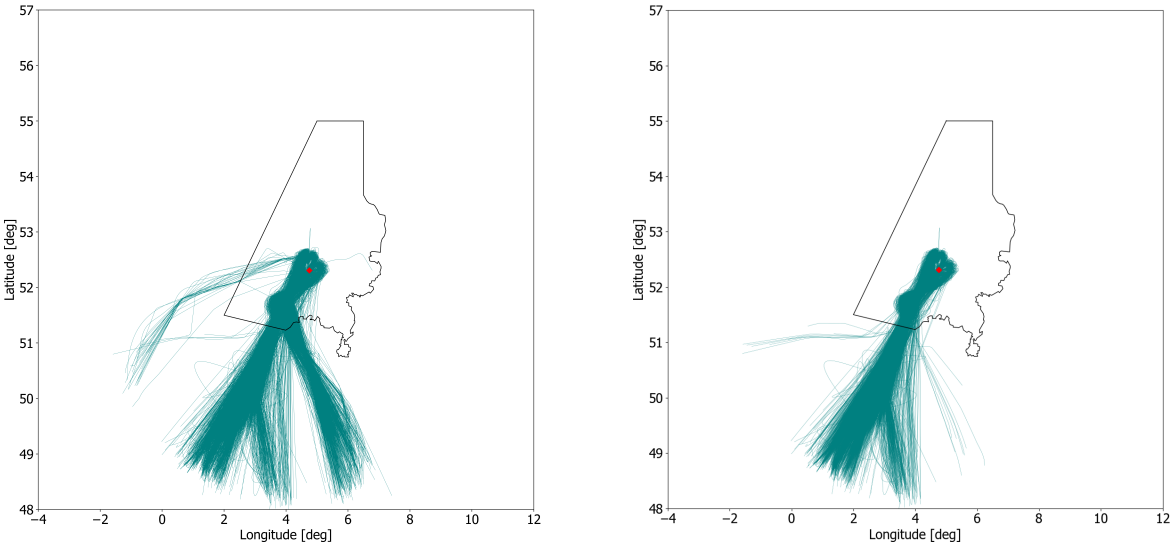
Single cluster results of STAR-based clustering

This appendix contains the final results of clustering using the STAR-based method on a single-cluster level. These results can also be found in the overview of all clusters in part I, but are provided here as they form the basis for the final experiments and results of part I and showing them separately allows for visual inspection of the quality of the clustering approach.

The number of trajectories per cluster can be found in table C.1. The results for the DENUT cluster, fig. C.1, have been shown in the scientific paper (part I) as an example. Here it was observed that the Hausdorff distance metric is not able to capture the details required for this trajectory set as it clustered trajectories from different routes together. Similar behaviour is observed in the HELEN, LAMSO, NARSO, REDFA and TOPPA clusters, see figures C.3, C.4, C.6, C.8 and C.10 respectively.

Table C.1: Number of trajectories per cluster, for both distance metrics and using the STAR-based clustering method.

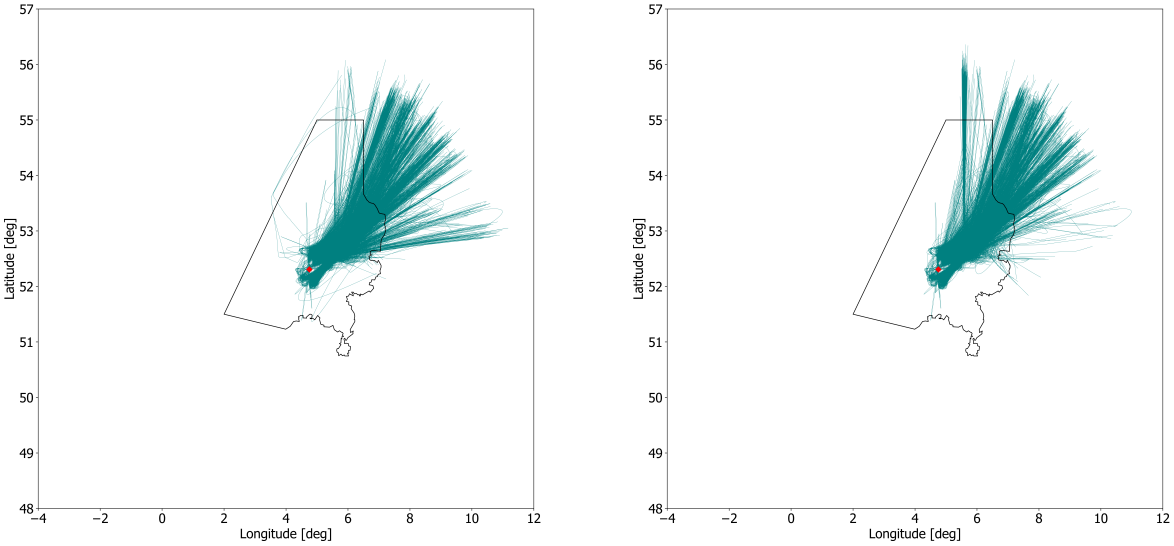
Cluster	**Hausdorff**	**SSPD**
DENUT	8234	6390
EELDE	4719	5746
HELEN	2031	3734
LAMSO	521	1253
MOLIX	3008	4475
NARSO	1767	1022
NORKU	9239	8665
REDFA	6997	5799
REKKEN	813	1366
TOPPA	2089	968



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

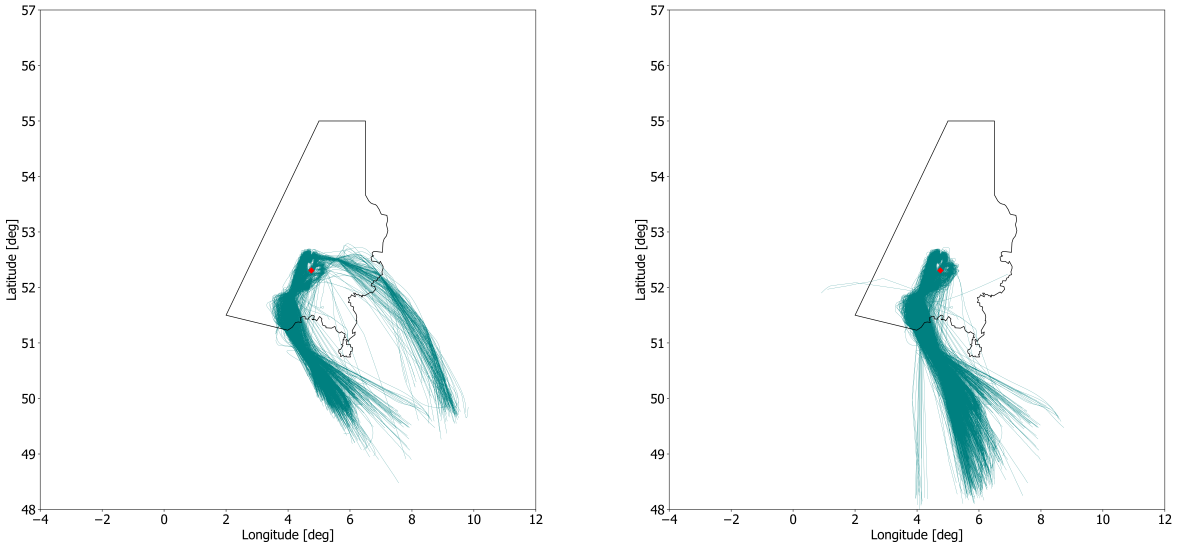
Figure C.1: DENUT cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

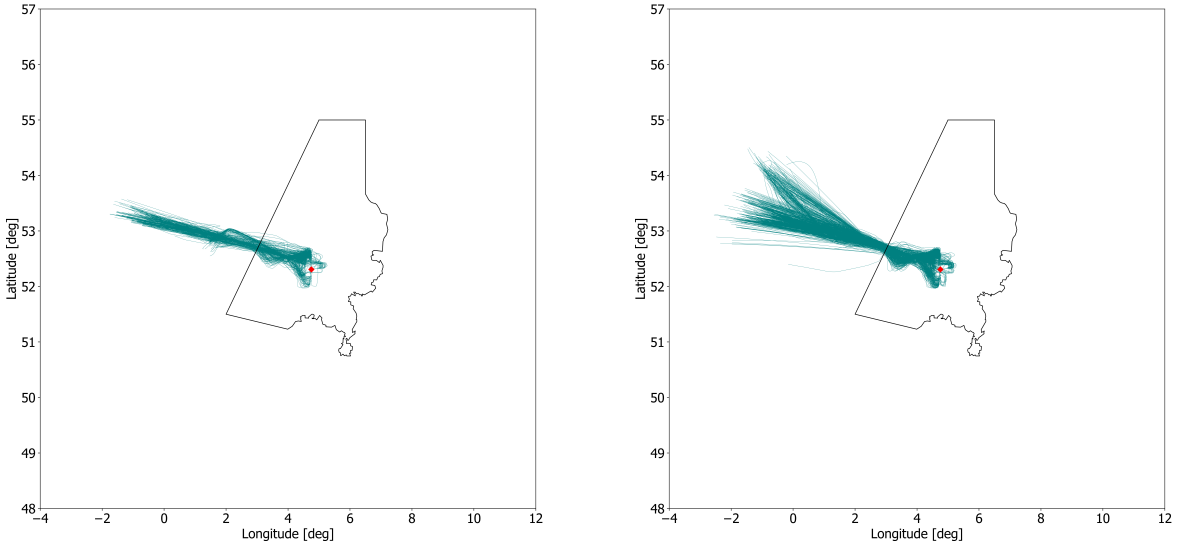
Figure C.2: EELDE cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

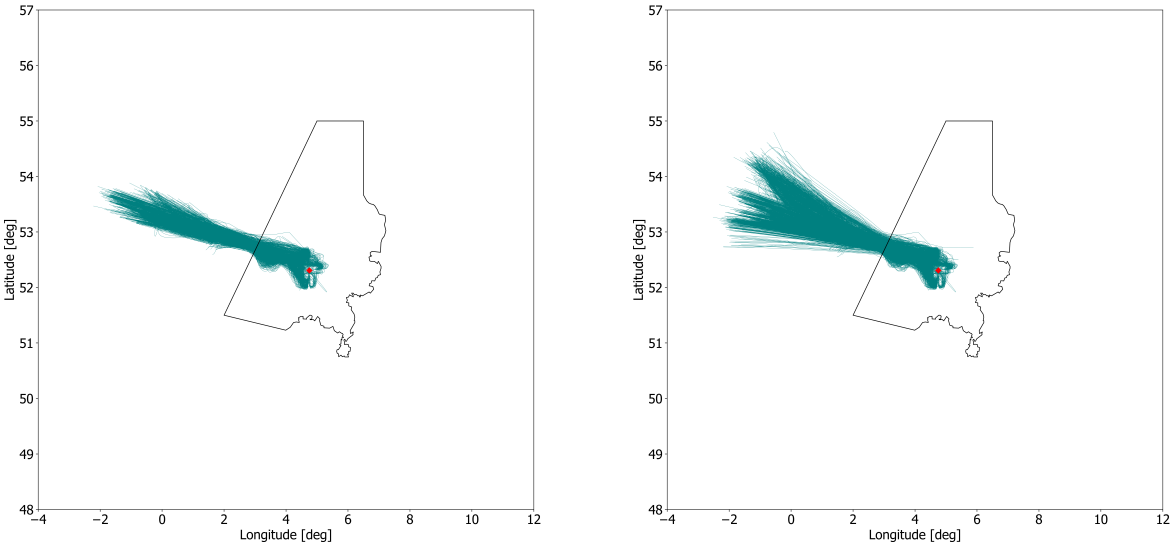
Figure C.3: HELEN cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

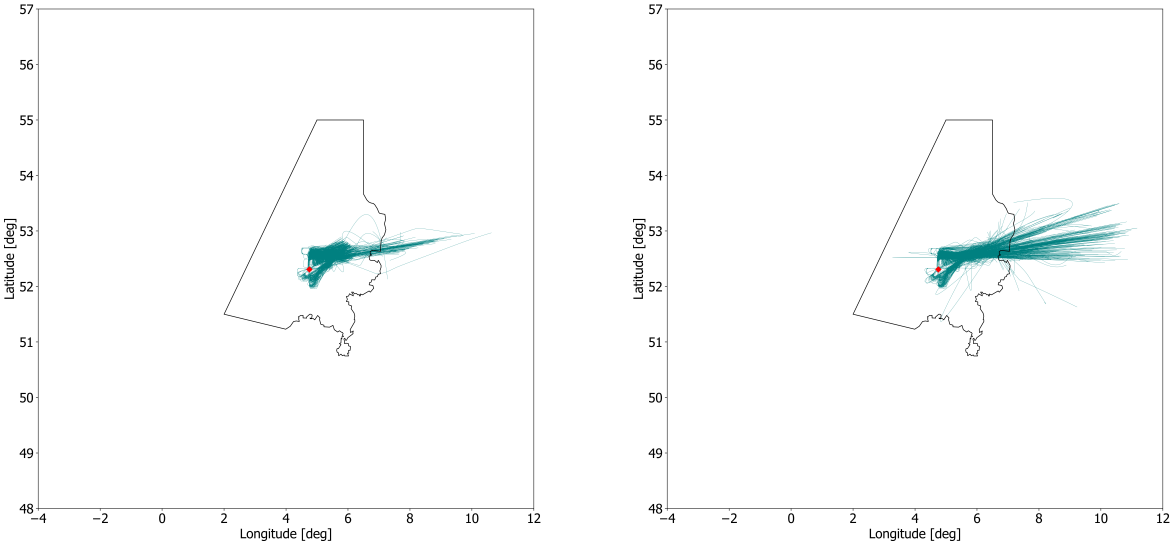
Figure C.4: LAMSO cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

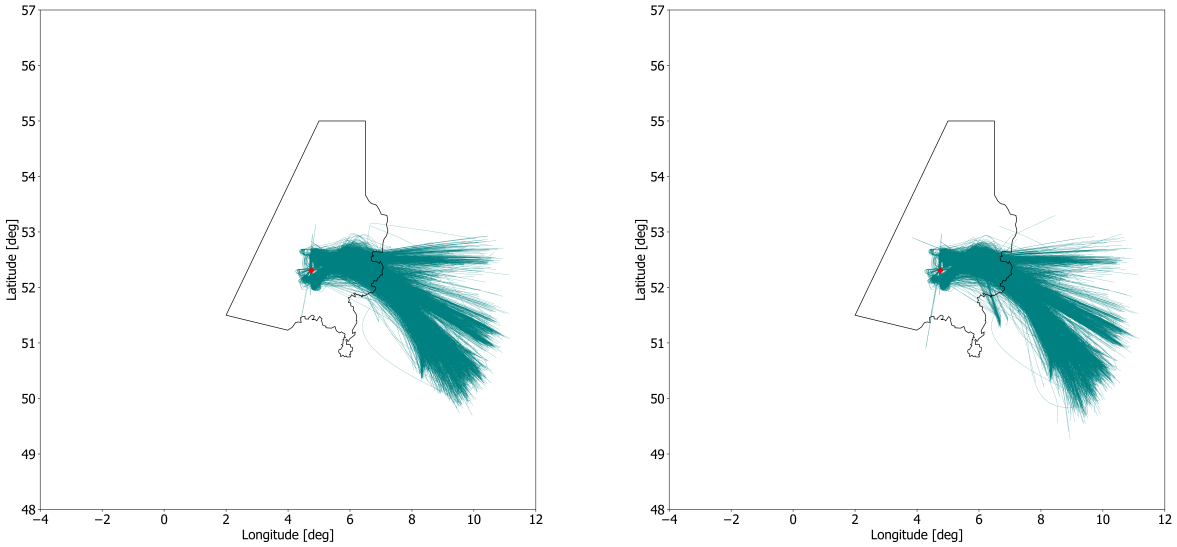
Figure C.5: MOLIX cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

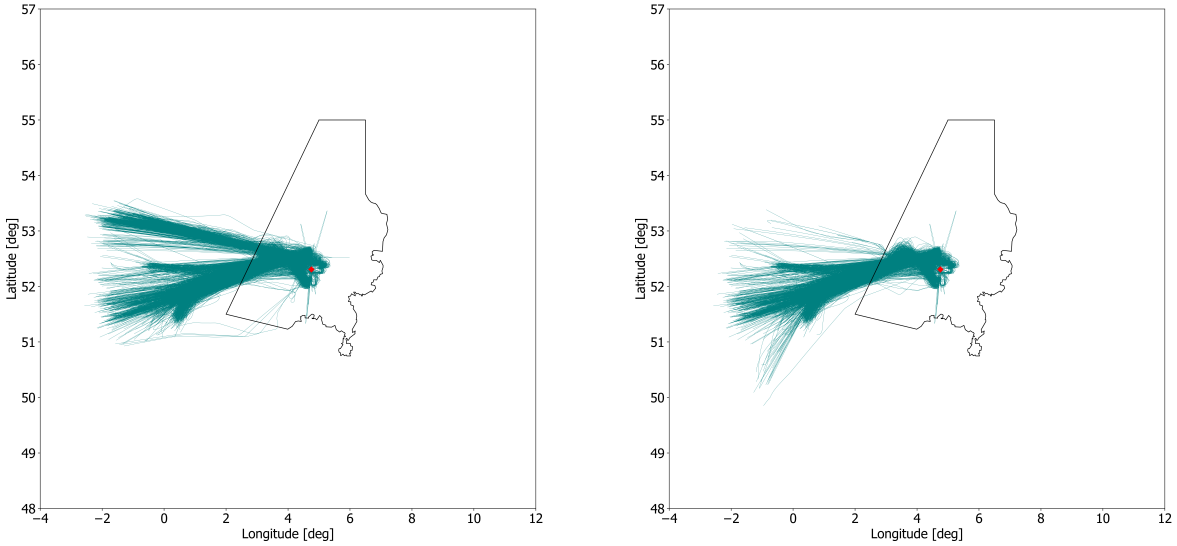
Figure C.6: NARSO cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

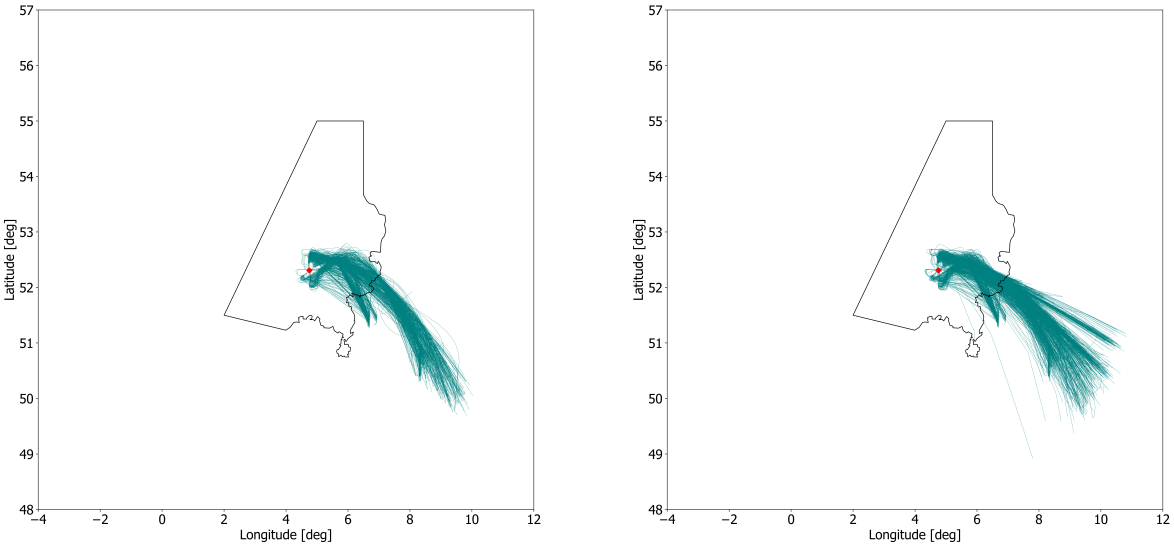
Figure C.7: NORKU cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

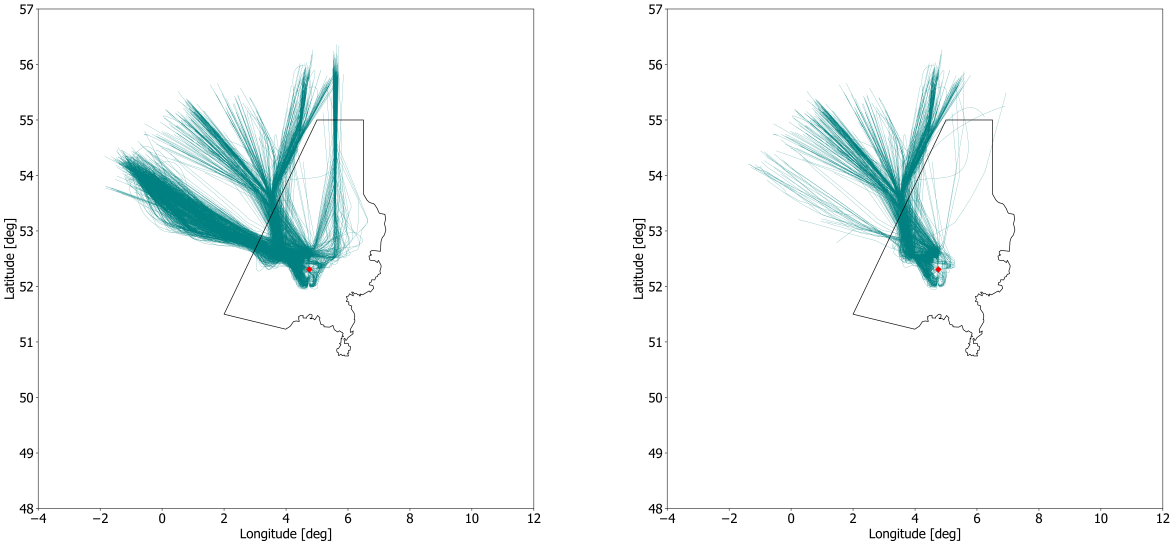
Figure C.8: REDFA cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

Figure C.9: REKKEN cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.



(a) Using Hausdorff distance metric.

(b) Using SSPD distance metric.

Figure C.10: TOPPA cluster, constructed using SSPD and Hausdorff distance metrics with the STAR-based clustering method. The black contour indicates the Dutch FIR and the red dot indicates EHAM.

D

Prediction results of all clusters

As explained in part I, the predictors were tuned to perform well on one cluster of trajectories, for which the LAMSO cluster is used. However, it is assumed that the model architecture as tuned for the LAMSO cluster is sufficient to provide good results on the other clusters as well. This is assumed as the type of data and the problem posed is similar, as well as the complexity of the relationships and patterns in the data. To show the validity of this assumption and to provide an overview of all results obtained in this study, the results of the rest of the clusters is presented here.

The amount of experiments performed and therefore results obtained in this study is large with ten clusters and four predictors. Therefore, the results are presented on an experiment basis instead of the distribution of the error over the time to landing as presented in part I. The three spatial metrics can be found in section D.1, section D.2 and section D.3. The time error can be found in section D.4. Finally, an overview of all results by means of the average and standard deviation of the error of all experiments is presented in section D.5.

D.1 Relative along track error

The relative along track error per experiment is shown in fig. D.1. It is observed that the relative performance of the four predictors is the same over all clusters. The GBM predictor outperforms the others, while the LSTM is second in terms of relative along track accuracy, the MLP is the worst performer of the novel predictors, while all novel predictors outperform the baseline simulations using APMs. Also, it is observed that the relative along track error is similar over the clusters.

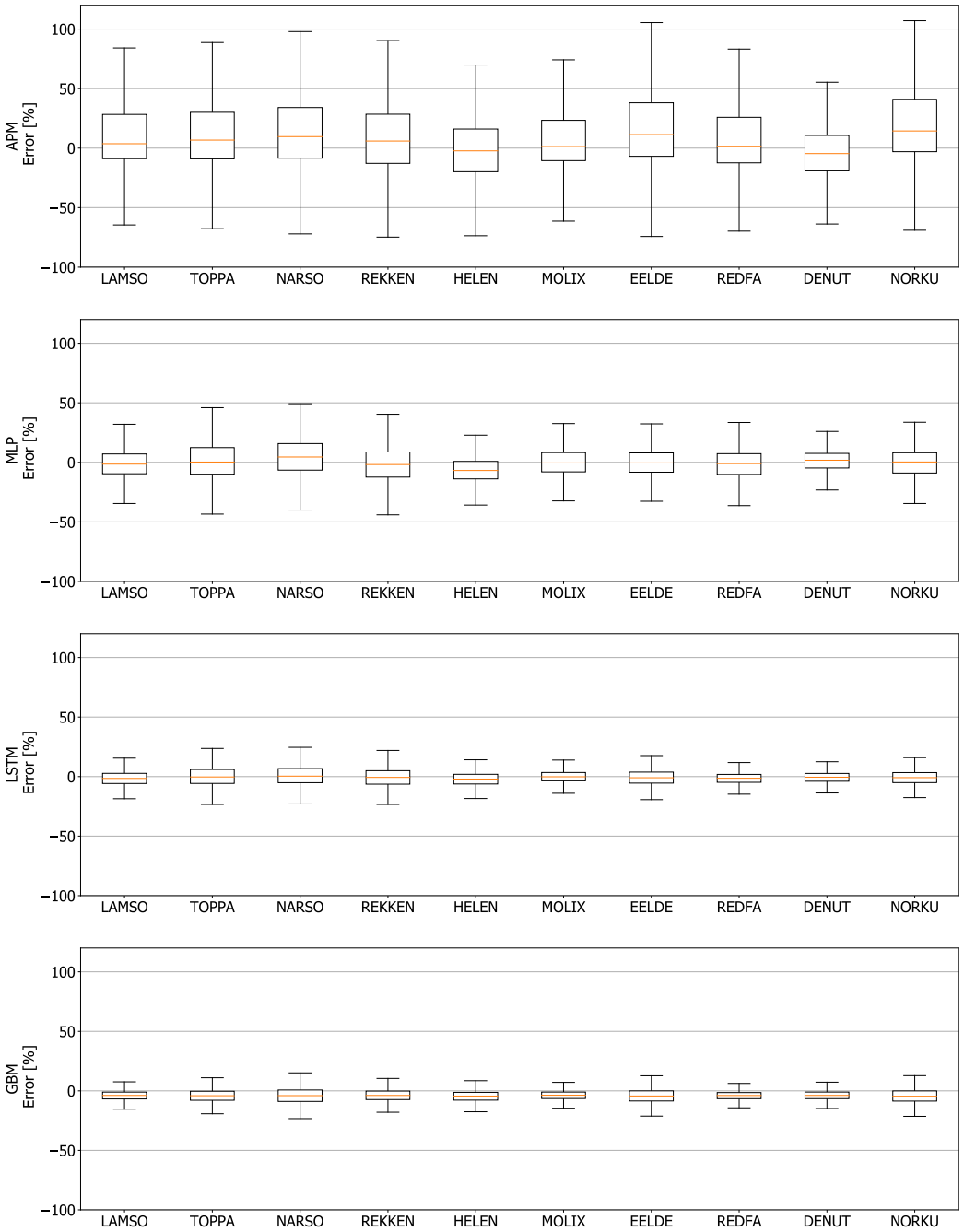


Figure D.1: Relative along track error of all experiments. Boxplots are shown for each predictor and cluster combination. The clusters are in order of increasing size, apart from the LAMSO cluster.

D.2 Cross track error

The cross track error per experiment is shown in fig. D.2. Again, it is observed that the relative performance of the four predictors is the same over all clusters. The difference between the performance of the novel predictors is less distinct, which is also the case in the results presented in part I. However, it is observed that the cross track error is similar over the clusters.

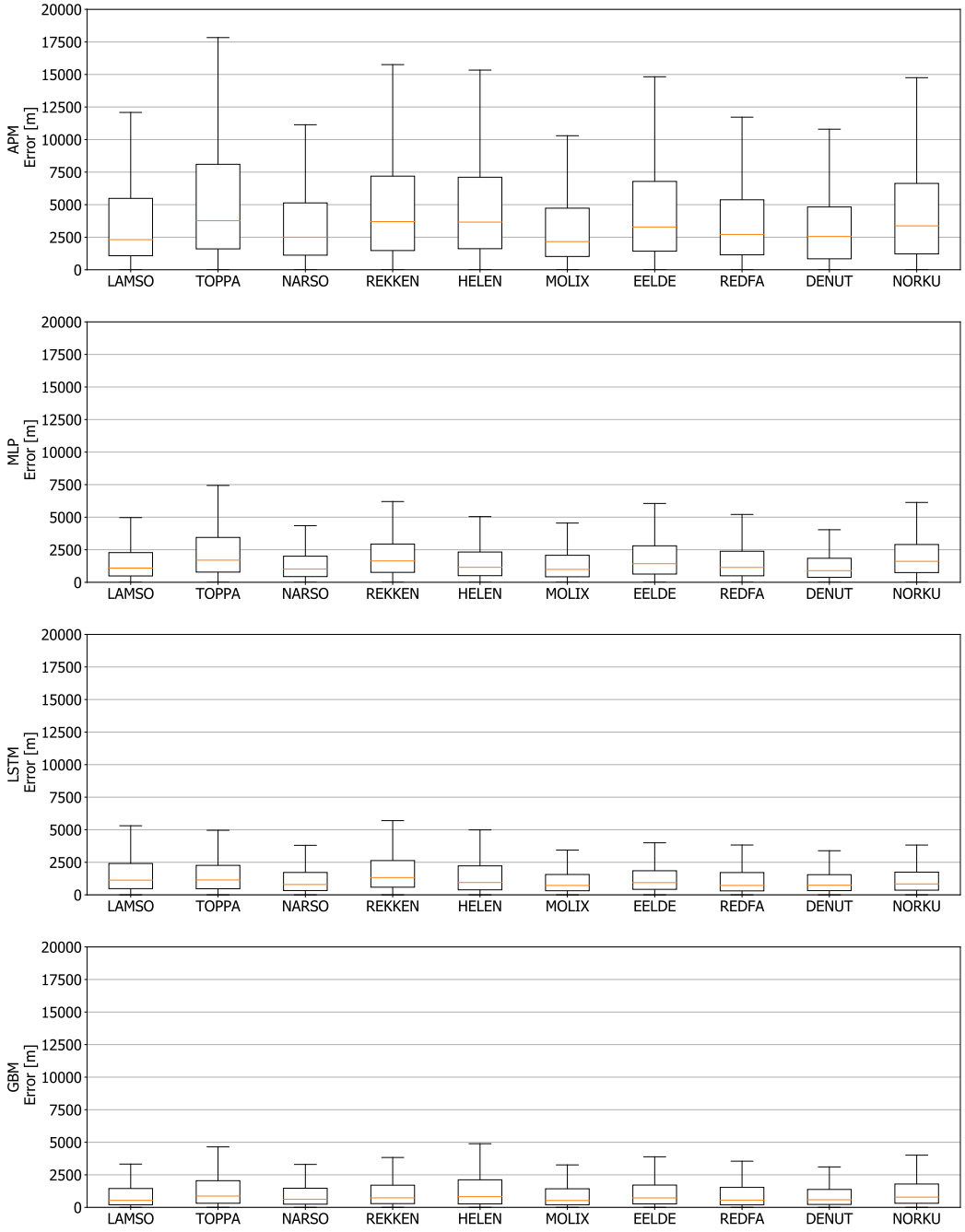


Figure D.2: Cross track error of all experiments. Boxplots are shown for each predictor and cluster combination. The clusters are in order of increasing size, apart from the LAMSO cluster.

D.3 Altitude error

The vertical or altitude error per experiment is shown in fig. D.3. It is observed that the relative performance of the four predictors is the same over all clusters. However, the LSTM and MLP predictors perform similar whereas the GBM predictor outperforms the other predictors, which confirms the results presented in part I. Finally, it is again observed that the altitude error is similar over the clusters.

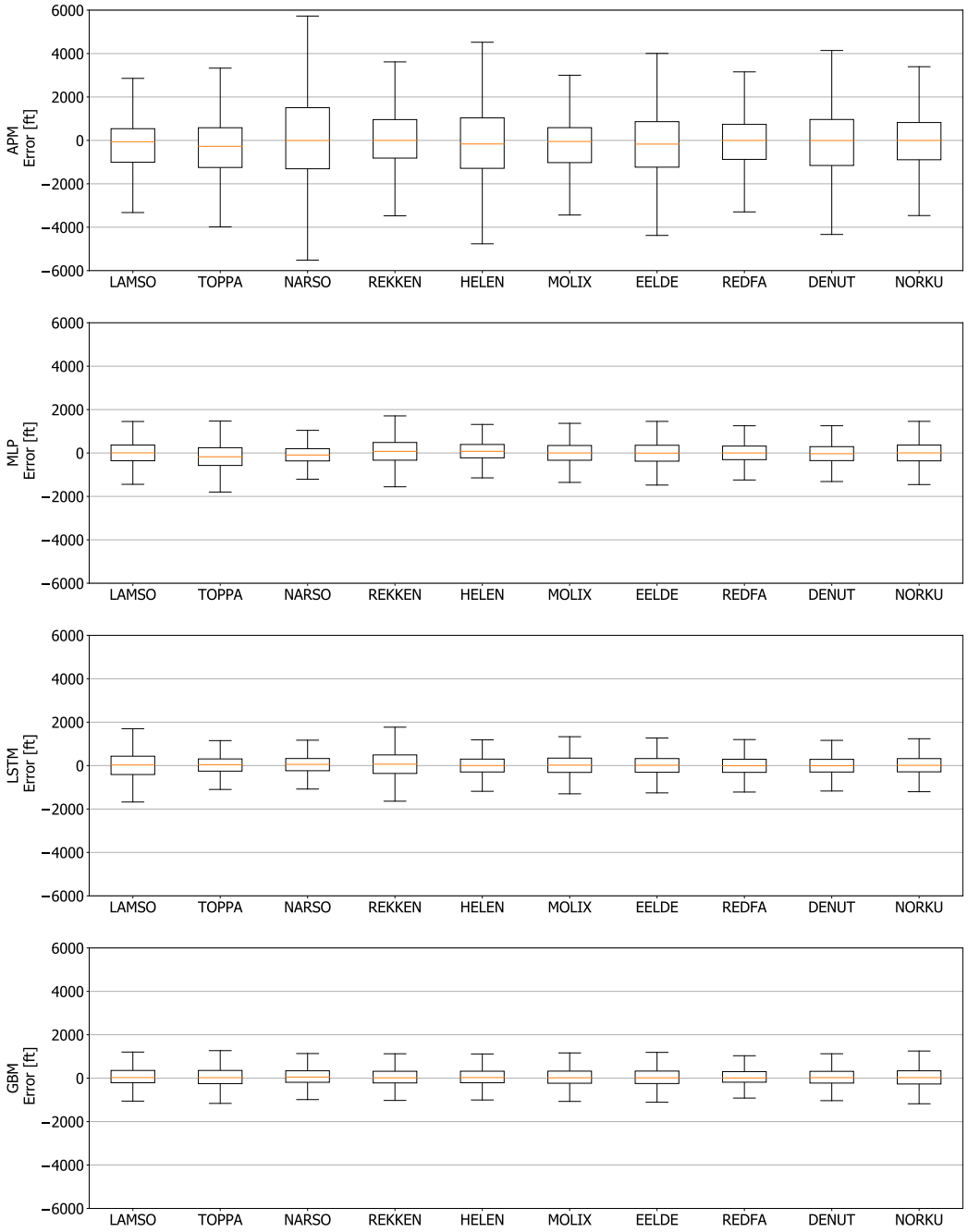


Figure D.3: Altitude error of all experiments. Boxplots are shown for each predictor and cluster combination. The clusters are in order of increasing size, apart from the LAMSO cluster.

D.4 Time error

The time error, ETA prediction, per experiment is shown in fig. D.4. It is observed that the MLP predictor shows worse performance compared to the other novel predictors. Apart from the LAMSO, NARSO, MOLIX and DENUT clusters, it also performs worse than the baseline simulations.

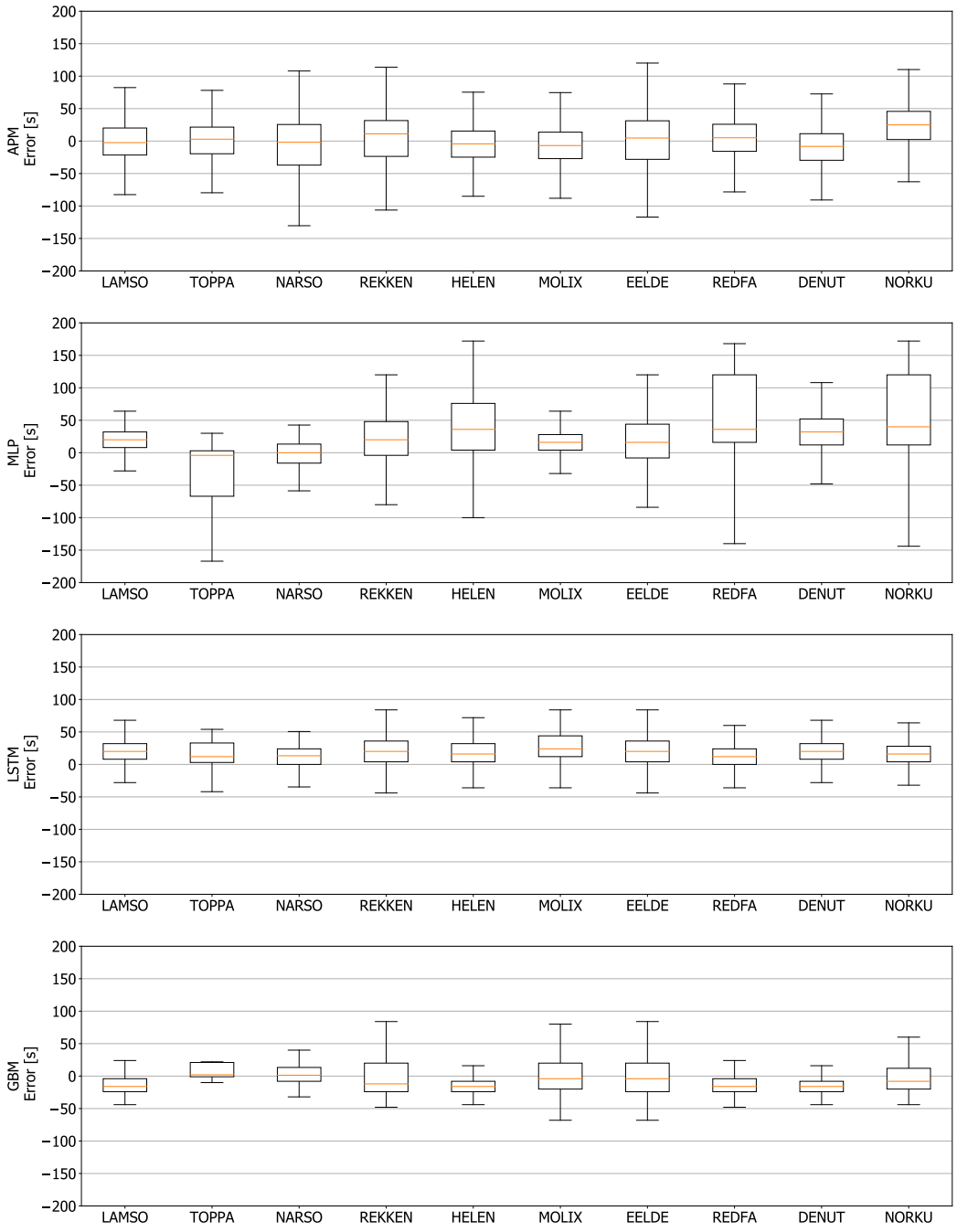


Figure D.4: Time error of all experiments. Boxplots are shown for each predictor and cluster combination. The clusters are in order of increasing size, apart from the LAMSO cluster.

D.5 Overview of all spatial and temporal evaluation metrics

All results of the experiments are presented in the previous sections. For completeness, in table D.1 the results of all these experiments are presented by means of the average and standard deviation of each error.

Overall, it is observed that similar results can be obtained when using the model architecture tuned on the LAMSO cluster for training and predicting using other clusters. In table D.1 and previous sections it is observed that the GBM predictor outperforms the others in all experiments on almost all metrics.

However, it is expected that tuning the models for each cluster separately would improve the results of all predictors, especially in terms of the ETA error. The LSTM predictor is found to be resilient against parameter tuning in part I, which indicates that it performs well with a relative wide range of parameter settings. Therefore, it is expected that tuning the LSTM predictor for each cluster separately would result in less improvements in comparison with the MLP and GBM predictors.

Table D.1: Overview of the results for all four models on all 10 clusters, showing the mean and standard deviation of each metric.

			along [m]		rel. alo. [%]		cross [m]		vertical [ft]		ETA [s]	
Cluster	**Size**	**Pred.**	**Mean**	**StD.**	**Mean**	**StD.**	**Mean**	**StD.**	**Mean**	**StD.**	**Mean**	**StD.**
LAMSO	1253	SIM	2184	11922	21	109	4805	7722	-16	1605	-1	33
LAMSO	1253	MLP	-543	3694	-1	31	1752	1993	12	751	2	116
LAMSO	1253	LSTM	-448	2798	-2	30	1925	2342	11	901	-2	131
LAMSO	1253	GBM	-944	1982	-5	19	1186	1714	72	661	-23	111
TOPPA	968	SIM	2261	10175	18	76	5998	7308	-103	1752	-2	53
TOPPA	968	MLP	94	12068	4	76	5011	7634	-224	1703	-35	177
TOPPA	968	LSTM	81	4917	1	31	2808	3385	16	1035	56	98
TOPPA	968	GBM	-917	2246	-4	11	1546	1970	39	694	12	159
NARSO	1022	SIM	2186	12522	24	309	4143	6745	157	2210	-32	128
NARSO	1022	MLP	830	7727	6	66	2609	3007	-170	1223	-23	214
NARSO	1022	LSTM	10	5904	2	48	2296	3131	29	1164	-8	167
NARSO	1022	GBM	-842	3255	-4	35	1183	1742	60	697	0	96
REKKEN	1366	SIM	1141	10107	14	92	5295	6093	35	1619	-17	109
REKKEN	1366	MLP	-523	4613	-1	32	2228	2540	79	932	-32	283
REKKEN	1366	LSTM	-236	3768	0	30	2016	2308	48	1033	-19	166
REKKEN	1366	GBM	-842	2482	-4	19	1290	1629	26	790	-30	150
HELEN	3734	SIM	-59	9471	7	93	5250	5735	-66	1659	-8	47
HELEN	3734	MLP	-1303	3336	-6	58	1809	2104	81	714	1	248
HELEN	3734	LSTM	-517	2459	-2	51	1776	2307	-9	790	-4	187
HELEN	3734	GBM	-856	5079	-4	32	1595	2181	57	661	-36	175
MOLIX	4475	SIM	1462	11029	17	99	4133	6696	9	1652	-7	51
MOLIX	4475	MLP	-302	3306	2	43	1635	1972	12	750	-13	168
MOLIX	4475	LSTM	-118	2501	0	35	1377	1955	0	771	4	170
MOLIX	4475	GBM	-841	1719	-4	55	1137	1613	44	715	-10	111
EELDE	5746	SIM	2744	11516	26	117	5258	6729	-70	1753	-12	103
EELDE	5746	MLP	-340	4001	1	63	2096	2419	-6	906	-19	185
EELDE	5746	LSTM	-339	3474	0	45	1485	1889	-6	870	-21	173
EELDE	5746	GBM	-895	2698	-4	20	1303	1703	52	704	-27	166
REDFA	5799	SIM	1593	29329	20	291	4746	26610	104	1588	1	146
REDFA	5799	MLP	-385	3782	-2	47	1778	2271	12	710	27	164
REDFA	5799	LSTM	-478	2876	-2	38	1461	2089	-9	809	-48	208
REDFA	5799	GBM	-962	2527	-5	27	1198	1713	54	640	-17	86
DENUT	6390	SIM	-583	9933	4	106	3825	5311	-49	1608	-10	35
DENUT	6390	MLP	137	3171	1	48	1475	1888	-25	808	7	171
DENUT	6390	LSTM	-182	2492	0	54	1277	1739	-11	794	0	159
DENUT	6390	GBM	-855	2166	-4	32	1131	1664	46	786	-29	88
NORKU	8665	SIM	3466	25127	30	298	5211	16125	-11	1609	21	65
NORKU	8665	MLP	-217	3844	0	50	2163	2148	9	902	20	184
NORKU	8665	LSTM	-302	2584	-1	39	1338	1623	13	723	-9	142
NORKU	8665	GBM	-897	2579	-4	34	1372	1757	27	787	-5	72

III

Preliminary Report [already graded]

1

Introduction

For the past century, the amount of people and goods that travel by means of flying has been increasing steadily. In the Netherlands for example, the main airport of both the country and its capital Amsterdam, Schiphol Airport (EHAM), has reached its current limit of 500,000 flights annually. This leads to a full airspace near busy airports where lots of aircraft are landing, taking off and passing at the same time. Air traffic controllers ensure safety and efficiency by monitoring and guiding the aircraft through the upper airspace, control areas and terminal control areas. In order to maintain safety, air traffic controllers (ATCo) can direct aircraft into holding patterns or tell them to slow down if their airspace becomes to busy or when no landing spot is available. At the same time, they can surpass the flight plan of an aircraft by providing a direct-to, a shortcut, when enough room is available and thereby making the flight more efficient.

In order to provide ATCo with accurate predictions of how busy a sector will be in the near future, trajectories of aircraft are predicted. Currently, this is done using aircraft performance parameters (APPs), which provide an indication of the typical performance of a specific type of aircraft in a certain flight phase. Using kinematic and dynamic models, the trajectories of the aircraft are predicted with these parameters as an input. These trajectories constitute of points in 3D space and time, which together form a so-called 4D trajectory. Although these predictions serve as a good estimate of a future position of an aircraft, several major indicators of trajectories, such as weather and other traffic, are often not taken into account in these models.

Over the past decades, efforts have been made to increase the information available on each flight and aircraft. All commercial aircraft currently broadcast 'ADS-B' data, which provides an accurate position and the speeds of the aircraft. This data can be received by anyone with an antenna and is therefore openly accessible. Also, increasingly accurate meteorology data, such as temperature and wind speeds, is becoming openly available.

Using this these two types of data, machine learning models can be trained on the historical flight data and then applied on current flights. These types of trajectory predictors take more information into account than the current type of predictors used today by ATCo and can therefore serve as a more effective tool. This could then in turn lead to more efficient and safe flying of aircraft as collisions and busy sectors can be identified earlier and with higher accuracy. Also, additional data such as the origin and destination of a flight can be added, enriching the data sets in order to make the models even more accurate. Research in this field has focused on either improving the current predictors using aircraft performance models (APMs) or developing new predictors, both using machine learning methods. Recent advancements show a potential for individual methods but a comparison of the best performing algorithms is missing. Also, most studies take different factors into account, limiting the comparability between them.

Research Questions

Considering the need for more accurate trajectory predictions for ATC in order to improve safety and efficiency in busy air spaces and the opportunities arising due to data availability and the available processing power nowadays, it is proposed to develop several machine learning trajectory prediction models and compare these to baseline methods using existing APMs and simulators. The research question will be as follows:

'Which machine learning algorithm is best suited to perform trajectory prediction for 4D trajectories of aircraft?'

The suitability of an algorithm can be determined using several measures. Seeing as a 4D trajectory prediction model is being analysed, a first indication of performance is to measure the time error and spatial errors. Furthermore, to support ATC the stability of the predictor is of importance, as well as the speed of the predictor. It should be noted that the stability of machine learning based predictors has not been investigated in recent literature yet. This implicates the following four sub questions:

1. *Which algorithm provides the most accurate predictions in terms of ETA?*
2. *Which algorithm provides the most accurate predictions in terms of cross- and along-track errors along the whole trajectory?*
3. *Which algorithm produces the most stable predictions in terms of change in ETA?*
4. *Which algorithm is favourable in terms of processing power, assessed both in the learning as in the prediction phase?*

Structure of the report

This report details the work performed in order to analyse which type of machine learning algorithm is best suited to perform trajectory prediction for 4D trajectories of aircraft. First, an extensive review of previous and current work in this direction will be presented in chapter 2. Then, the data used in this study will be shown in chapter 3, where also the processing and an analysis of the data will be explained. The experiment setup will be presented in chapter 4 with an accompanying planning in chapter 5. Finally, conclusions will be drawn in chapter 6.

2

Literature Review

In this chapter background information on the topic and relevant recent efforts in the field of trajectory prediction (TP) will be presented. Following the brief description on trajectory prediction in the introduction, several different forms of TP will be explained in detail in section 2.1. Then, a more elaborate introduction to the chosen machine learning algorithms will be given in section 2.2, after which the inputs and outputs of the to be built models will be introduced in section 2.3. The measures with which predictors are to be evaluated are presented in section 2.4.

2.1 Trajectory prediction methods

As explained in the introduction, trajectory prediction of aircraft is used to estimate the amount of traffic in a sector of airspace in a future point in time and to predict collisions of aircraft. In order to do so, several methods have been developed. The main group of predictors consist of a kinematic model with as input a set of aircraft performance parameters. These models determine the aircraft states and how these propagate over time, forming a trajectory. A lot of research has been performed in this field, which is detailed in subsection 2.1.1. Then, a more recent development entails the same principal, but with variable APPs as input. This is detailed in subsection 2.1.2. Then, a method in which historical flight data is used to directly predict trajectories by means of machine learning models is presented in subsection 2.1.3. A similar method entails first clustering similar trajectories, after which these clusters serve as a basis to train individual machine learning models. This is presented in section section 2.1.4, after which the best performing algorithms from previous research are chosen in section section 2.1.6. The technical details of the algorithms and methods presented in this section will be shown in section section 2.2.

2.1.1 Conventional trajectory prediction using aircraft performance models

The current standard for trajectory prediction in ATM are systems using APMs, which are based on the Total Energy Model (TEM). The TEM sets the rate of work as performed by the forces acting on the aircraft, for example the thrust forces applied by the engines, equal to the rate of potential and kinetic energy [49]. In BADA [50] as well as in OpenAP [55], this is done using a combination of kinematic and dynamic aircraft performance models which simulate the behaviour of aircraft under given circumstances. Kinematic models simulate the motion of an aircraft without considering the forces acting on the aircraft, while dynamic models for this purpose are generally point-mass models with forces working on the centre of gravity. These models describing the aircraft's motion are combined

with flight plan or intent data in order to predict the trajectory of an aircraft and inputs are the aircraft and engine properties.

Although these models provide accurate predictions of aircraft behaviour on set trajectories or during flight phases such as climb and descent, they cannot predict the route an aircraft will fly. Therefore, they are dependent on flight plan data or similar sources which do not take any circumstances such as busy sectors and other traffic into account. Flight plans are often only executed in broad lines rather than to the detail of each waypoint, especially in the final part of a flight, preventing predictors using APMs to be very accurate on the long term. It should be noted, however, that the current ATM structure is using these methods and therefore improving the existing methods rather than developing new ones will simplify updating the models when it comes to implementation of new solutions.

2.1.2 Trajectory Prediction using adjusted aircraft performance models

The APMs detailed in the previous section can also be adjusted to the specific flight at hand. Two methods have been presented in recent literature. First of all, the way in which APPs, the inputs to the APMs, are chosen from a database of APPs can be enhanced. These are chosen based on the aircraft type, flight phase, but also on more specific flight parameters, such as the current mass and the CAS and Mach numbers. Alligier et al. [1] proposed a least-squares method on the thrust law to estimate the mass of an aircraft to use this as input to BADA. Later, in [3] and [4] they propose to estimate the mass and the CAS & Mach numbers using machine learning methods, which are then used as input to BADA. Hrastovec et al. [31] propose a similar machine learning method, also estimating mass and speeds. Hadjaz et al. [22] tuned existing APPs using a 'Covariance Matrix Adaptation Evolution Strategy', which entails adapting the APPs by minimising the difference between the predicted and true trajectories. However, the accuracy of the latter deteriorates for predictions of 10 minutes and further due to over fitting, which is defined as an algorithm being trained so well on the trainin data that its performance deteriorates on new test data.

A similar strategy is found in research which proposes to estimate new APPs, without having to estimate inputs to existing APP models first. The advantage of this method is that using historical flight data more specific sets of APPs can be estimated, tailored to specific routes and circumstances, which leads to more accurate results. Hrastovec et al. [30] proposes a method using a 'Nearest Neighbours' algorithm to predict APPs, and in [29] they add several regression methods in this comparison.

The main advantage of these methods is the compatibility with the current type of ATM and ATC systems. The new sets of APPs can be used in current predictors without having to change infrastructure or integration with the current systems. Also, it is a fairly robust method if information is missing, as APPs from for example BADA and OpenAP can still be used with the existing methods. However, the disadvantage is that no full route is predicted and flight plans will still be necessary. Also, data used to determine these new APPs can also be used to perform direct predictions, which is detailed in the next section.

2.1.3 Trajectory prediction using direct machine learning models

With the increasing amount of data available on flights and the increasing amount of processing power available, a new solution to trajectory prediction became feasible several years ago. By using historical flight- and other data to train machine learning models, trajectories can be predicted directly without having to estimate APPs first and then running a simulation with kinematic and dynamic models. This method eliminates all conventional physical calculations from the predictions and has proven to work for specific use cases.

In 1999, Le Fablec et al. [33] proposed to predict the vertical trajectory of aircraft using a neural network and was one of the first to perform such an attempt. Similarly, Cheng et al. [13] proposed a

traffic flow predictor using a neural network. Both use a single-layer neural network. This line of work was also applied on delay prediction, which serves a similar purpose but does not entail the full analysis needed for trajectory prediction. This was done by Takeichi et al. [56].

Later, Liu et al. [36] and Pang et al. [42] propose an approach using multi-layered recurrent neural networks. An example of these are 'Long Short Term Memory' neural networks with convolutional layers. These are better at predicting time-varying data compared to feedforward neural networks and with a 'deep' neural network structure with two embedded convolutional layers, these models outperform simple shallow neural networks as proposed by the earlier work in [33], [13].

In 2013, another approach was initiated by de Leege et al. [15]. They propose supervised regression techniques such as 'Support Vector Regression' and 'Generalized Linear Models', which are compared to (simple) neural networks. The regression methods perform similar, but slightly better than the neural network. Similarly, Hamed et al. [23] and Alligier et al. [5] propose several regression methods which outperform the more simple neural networks. Especially 'Gradient Boosting Machines' and 'Multivariate Locally Weighted Linear Regression' outperform the other methods. Tastambekov et al. [57] propose functional regression, in which the data is fitted on a function. However, too many parameters leads to over fitting and a high sensitivity to noise and variation in the input parameters.

A major benefit of these methods is that any type of data which is indicative of aircraft performance and traffic flow can be taken into account and used to train the models. However, with the amount of data increasing both in terms of features and variables as in terms of amount of historical data, the models will become fairly large. A solution to these problems is presented in the next section.

2.1.4 Trajectory prediction using clustering and direct machine learning models

Similar to the methods presented in the previous section, trajectory prediction models can also be trained on clustered trajectories. In order to do so, trajectories are clustered using either pre-determined trajectory characteristics such as origin and destination, or using machine learning methods which are able to determine the basis for clustering themselves. After clustering the machine learning models are trained for each specific cluster, which leads to more efficient training and prediction. Each new flight needs to be assigned to a cluster before the prediction can take place. The machine learning models applied after clustering can be the same as presented before in the direct method. First, a brief overview will be given of relevant literature for clustering of trajectories only, after which literature in which the clustering and prediction combination is tested will be presented.

Clustering of data is a typical purpose for machine learning algorithms and several approaches to clustering aircraft trajectories have been explored in the past. A first attempt in this direction was performed by Jesse et al. [32], which provides an overview of three different categories, namely partitional, density based and hierarchical clustering. They concluded that density based clustering in combination with fuzzy memberships provides the most accurate results for the dataset at hand, but that the clustering process has to be faster. In 2010, Rehm et al. [44] presented a hierarchical clustering method for aircraft trajectories. In 2011, Leiden et al. [34] investigated several density based methods with one day of traffic data and concluded that for ridge detection and density based spatial clustering for applications with noise the data was to sparse and k-means clustering proved to be the best performing method in this case. However, when larger datasets are used, DBSCAN is often found to be better than other density-based methods. Gariel et al. [21] has shown that DBSCAN works better than k-means due to the noise which is inherently present in trajectory data and Salaun et al. [48] proposed a categorization using hierarchical clustering to split the trajectories based on the destinations, but uses DBSCAN to form clusters within these categories. Then, Enriquez et al. [16] proposed a spectral clustering method for aircraft trajectories, but Condé et al. [14] have shown that density based methods such as DBSCAN perform better. From this literature it was concluded that density based algorithms such as DBSCAN are preferred over distance based clustering methods as the shape of the clusters can be arbitrary. Also, DBSCAN performs well with noise and in the detection of outliers, which are inherently present in real traffic data. However, clustering can be a time consuming process, especially as a dataset is large.

Therefore, it should be investigated whether this behaviour can be mitigated.

Several studies perform dimensionality reduction using Principal Component Analysis (PCA) in order to speed up the clustering process. Nicol et al. [40] studied the implementation of functional PCA on aircraft trajectories and it was applied by both Gariel et al. [21] and Salaun et al. [48] before clustering using DBSCAN and it proved to speed up the clustering process.

With this research on clustering of trajectories in mind, literature using both clustering and trajectory prediction methods are now presented. Hong et al. [28] uses 'Dynamic Time Warping' for clustering and then applies a Multiple-Linear Regression (MLR) model on each cluster for prediction. After, Wang et al. [59], [60] performed clustering using DBSCAN after PCA. They compared a multi-cells neural network (MCNN) with the MLR, and concluded that the MCNN performs better. Later, they perform a clustering method based only on the runway usage and predict using a deep neural network with three hidden layers [61], however in this research only the ETA is predicted. Similar to Wang et al., Liu et al. [36] and Verdonk Gallego et al. [58] use PCA and DBSCAN for clustering, where this also proves to be an effective method.

Barrat et al. [8] apply a different clustering method. First, they extrapolate and shorten trajectories such that they are all of the same length, after which the similarity of the trajectories is calculated using the Euclidean norm. This is then used for k-means clustering. Each cluster has a mean trajectory, which serves as the prediction. New flights are coupled to these clusters using 'Gaussian Mixture Models', which assigns a trajectory to a cluster based on a certain probability. However, this method can introduce errors due to elongating and shortening the trajectories and the number of clusters has to be determined by hand.

2.1.5 Hybrid trajectory prediction using APMs and direct machine learning

A hybrid version, which has to our knowledge not been proposed before, can consist of direct machine learning methods to predict the trajectory in a rough spatial manner, while an APM is used to determine how the aircraft would fly through this 'flight plan'. This would come down to predicting the waypoints and altitudes, however in more detail to also cover direct-to's and diversions, and then executing this 'flight plan' using physics. The benefit of investigating this type of predictor would be that the performance of using physics versus historical data to predict the performance of the aircraft can be compared easily, as a full prediction including the time component will also be available from the direct machine learning method. Also this method may be easier to implement in the current ATM system.

2.1.6 Comparison of trajectory prediction methods

The five different lines of work presented in the previous sections can be compared on a scale of 'physics' to 'black-box approximation'. The conventional trajectory prediction method, using APPs, as introduced in section 2.1.1, is considered the most physical method available and represents the physical behaviour of aircraft directly. The APPs are determined using knowledge of the aircraft and aircraft flight envelope testing and some data originates from the aircraft manufacturers itself. Also, the kinematic models used to simulate the trajectories are simplified representations of actual aircraft. The second method, introduced in section 2.1.2, using adjusted APPs, is considered to be less supported by physics. This is due to the APPs being derived from historical flight data and other means and therefore finding the values for the APPs which fit best with the behaviour seen in aircraft. These values are still used in kinematic models which represent aircraft and they should still lie within the limits set for aircraft, but the values itself often do not originate from physical equations or tests. Here, the APPs are determined using a black-box approximation, while the APMs are still physical.

Then, the third and fourth methods, detailed in sections 2.1.3 and 2.1.4, are methods which are entirely black-box approximations. No physical quantities or calculations are used in these models and predictions, although they are based on historical flight data which in itself is based on physics. On the scale from physics to black-box approximation the fifth, hybrid, method would be located in the middle of the sequence.

For a complete overview on which type of predictor is best suited to perform 4D trajectory predictions, a comparison should be made between these five methods based on common inputs and outputs. For the sake of time and to limit the extent of this research, however, it is chosen to develop a baseline which represents the systems currently in use in the ATM system. The baseline is then compared with a more advanced method. This baseline entails conventional prediction using an APM such as BADA or OpenAP, after generating rough flight plans through the Dutch FIR. The more advanced methods will be based on the fourth method, by first clustering and then training a machine learning model on these clusters. Finally, both methods can also be compared with the actual flown trajectories. The fourth method is chosen as it provides the most promising and best performing results in recent literature.

For the advanced method, algorithms have to be chosen to compare. First of all, it is concluded the density based clustering using DBSCAN, after dimensionality reduction using PCA is expected to provide the best results. Then, for the machine learning algorithms used in the predictor several categories can be distinguished. From this literature review it was concluded that among the Decision Tree models, Gradient Boosting Models outperformed other DT methods, except for Random Forest in one case. Among the regression methods the simple linear and logistic regression methods, as well as Support Vector Machines, performed worse than the more sophisticated methods, Multivariate Locally Weighted Linear Regression (LWLR) being the top performer. Furthermore, it was found that most methods outperform simple, one-layer neural networks. However, deep neural networks have proven to perform better. A distinction can be made between deep feedforward networks and deep recurrent networks. The former often use Multi-layer Perceptrons (MLPs), while for the latter Long Short-term Memory (LSTM) networks, with or without convolutional layers, provide the best results.

Concluding, it is chosen to compare GBM, LWLR, MLPs and LSTMs as prediction algorithms in the fourth method, in combination with clustering using PCA and DBSCAN. The technical details of these algorithms will be introduced in the next section.

2.2 Machine learning algorithm specifications

Most models and algorithms proposed in the previous section use machine learning techniques. These techniques learn from historical data in order to predict future data. This learning can be done in various ways and choosing the correct machine learning algorithm depends on the type of data and the application or goals of the model. In the previous section the best performing algorithms from literature have been identified and categorised. Here, the specifications of the algorithms will be explained in detail and compared with other relevant state-of-the art algorithms.

Machine learning methods relevant for this research can be divided in three groups. Namely, relatively simple regression methods, more advanced regression methods which combine multiple steps and deep neural networks. All these methods can be applied on similar regression and classification problems, but differ in approach or structure. The simple regression methods are equations which are fitted on the data and are detailed in section 2.2.1. Then, some more advanced methods in which multiple simple regression methods are combined are presented in section 2.2.2. The previously identified algorithms GBM and LWLR belong to this category. Then, several deep neural networks are introduced. The latter includes the MLP and LSTM networks introduced in the previous section and these are presented in section 2.2.3.

2.2.1 Simple regression methods

Most machine learning methods rely on regression in one way or another. The basis for this is linear regression, which is therefore detailed shortly. In linear regression a target value (y) is predicted based on independent or explanatory variables (x). The accuracy of linear regression is assessed using a cost function and the regression is optimised using for example gradient descent. An example of such a model can be found in eq. (2.1).

$$\begin{aligned} \textit{Linear Regression}: \quad \hat{y}_i &= a_0 \cdot x_i + a_1 \\ \textit{Cost Function}: \quad C &= \frac{1}{n} \cdot \sum_{i=0}^{n} (y_i - \hat{y}_i)^2 \\ \textit{Gradient Descent}: \quad a_0 &= a_0 - LR \cdot \frac{dC}{da_0} \qquad a_1 = \quad a_1 - LR \cdot \frac{dC}{da_1} \end{aligned} \tag{2.1}$$

In this model LR is the learning rate, which is a parameter that defines how fast an algorithm learns and therefore has influence both on the convergence of the algorithm as on the overshoot. Every epoch, which is a run over all historical data, the cost function is calculated and using gradient descent the error in the cost function is minimised, resulting in a new linear regression equation which approximates the training data better. Algorithms using the same techniques but different functions include polynomial, least-squares and logistic regression. Furthermore, algorithms using multiple explanatory variables (x_1, x_2, ... , x_i) are called Multiple-Linear Regression (MLR) algorithms and operate in a similar way. Some simple regression methods have been analysed in previous research, often as a baseline for more advanced methods, by which they were always outperformed. Therefore they are not considered for further analysis as stand-alone methods. However, they form the basis of the more advanced methods, which are presented in the next section.

2.2.2 Advanced regression methods

More elaborate methods use simple regression as a basis but perform other actions on the data as well. Two forms used in previous research are 'Locally Weighted Linear Regression' (LWLR) and 'Locally Weighted Polynomial Regression' (LWPR). These are non-parametric regression methods, meaning that they do not learn a fixed set of parameters, but define this number themselves iteratively. The algorithms first define subsets of data using the Eucledian Distance, which is the perpendicular distance, after which either linear or polynomial regression is applied on each subset. By defining these subsets these algorithms can handle non-linear data, which is not possible using linear regression. This method was used by amongst others Hamed et al. [23].

Decision trees

Another set of advanced regression methods are decision trees (DTs), of which GBM are a version. DTs split the data into smaller groups based on the features of the data until the groups are of such size that labels can be assigned. They are used to classify and to perform regression, in which case they are called classification or regression trees [12]. The layout of a decision tree is presented in fig. 2.1. The internal nodes represent attributes or features, while the leafs represent the outcomes of the algorithm. The branches represent the rules and decisions leading from the input to the output. DTs can be binary, in which each split results in two new nodes, or multiway, in which each split can result in more than 3 new nodes. In the latter case, the trees are often more shallow. DTs are prone to overfitting, which means that the training data is learned so well that a model has problems adapting to new test data. In DTs this is prevented by means of 'pruning', which entails the process of removing sections of the tree that have little predictive power. Algorithms for DTs have large differences due to various

methods for splitting and pruning. An algorithm used often in the case of trajectory prediction ([60], [18]) is 'Classification and Regression Trees' (CART) [12], which generates multiple trees and chooses the optimal one. Splitting is performed by minimising the least squares deviation. However, DTs have disadvantages, such as a high variance, meaning a small change in data can lead to very different splits, overfitting and they look for local optima rather than global optima.

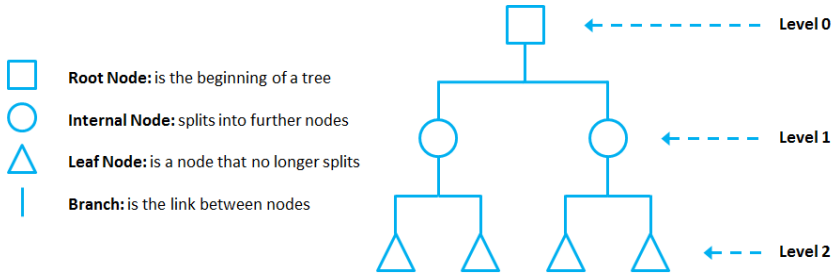


Figure 2.1: Layout of a shallow decision tree. Source:[1]

A solution to these disadvantages are ensemble methods, which combine multiple DTs to improve the performance. The first method in this field is Bootstrap Aggregation [11], which creates parallel subsets of training data and each subset is used to train DTs. The average of all predictions is chosen, leading to a more robust result. An advanced version on this is Random Forests (RF), which also creates random subsets of the features used to build the DTs. RF is used in various TP research efforts ([29], [36]) and often came out as one of the top performers.

A second variation are Gradient Boosting Models (GBM), proposed by Friedman [19]. It is used by Alligier et al. [3], [4], [5] and Liu et al. [36] and always provided the top results. GBM is a sequential method which fits consecutive DTs and at every step the errors of the previous tree are reduced. It has a structure similar to linear regression, with three elements:

- **Weak Learner:** shallow or short regression trees which are not capable of capturing the whole problem at once
- **Loss Function:** a differentiable function that defines the error of the weak learner with respect to the desired outcome
- **Additive Model:** a new tree is added after every iteration, improving the past one by using functional gradient descent: the weak learner DT is parameterized after calculating the loss function, these parameters are adapted to reduce the loss and these adapted parameters are then used to build a new tree. Existing trees are not changed.

Even after this procedure, GBMs still tend to overfit and are therefore regulated using constraints on the number of layers and nodes, weighted updates are applied, in which every new tree has a weighed contribution to the total outcome. A method proposed by Friedman [20] is 'Stochastic Gradient Boosting' and entails creating trees from sub samples of the training set. This reduces the correlation between the trees in the sequence and thereby reduces the over fitting on the training data.

2.2.3 Deep neural networks

With the top performers in the regression and decision tree domain identified, the next step is to detail the deep neural networks. As stated in section 2.1, two deep neural networks have been chosen to perform trajectory prediction with. The first method is a deep feed forward neural network, also called a deep Multi-Layer Perceptron network (MLP), while the other method is a recurrent neural network, a LSTM. Deep feed forward neural networks or MLPs are an extension of normal neural networks, which will be detailed first, after which recurrent neural nets and LSTMs will be presented.

[1] https://miro.medium.com/max/889/0IS9xKHt83nuERC9P

Feed forward neural networks

The structure of a neural network is shown in fig. 2.2 and consists of an input layer, hidden layers which perform calculations and an output layer. Perceptrons are also called neurons, hence the usage of the different names. The neurons in this neural network are computational units and each neuron operates in the following manner:

- **Inputs:** a set of inputs is received from either the input layer or a previous hidden layer.
- **Bias:** each neuron has a bias, which can be thought of as an extra input.
- **Weights:** each input and bias is weighted, similar to the coefficients in a regression function.
- **Activation or transfer function:** The weighted inputs + bias are summed and passed to a transfer function such as a sigmoid or tanh function, which maps the weighted input to the output. By using such a transfer function a threshold is applied on the neuron and the strength of the output is regulated.
- **Outputs:** each neuron passes an output to each neuron of the next hidden layer or, in case it is the output layer, to the final result.

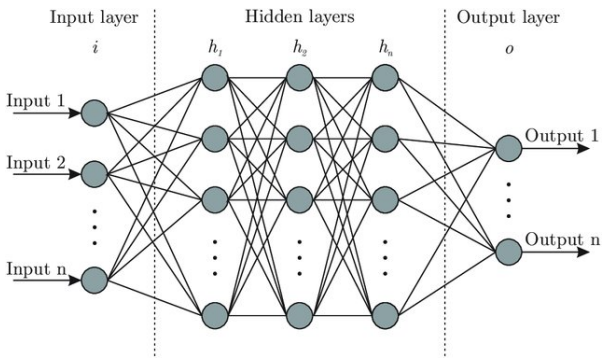


Figure 2.2: Structure of an artificial neural network including the input, hidden and output layers. Source: [10]

MLPs or deep feedforward neural networks are an extension to most neural networks used in previous trajectory prediction research, which usually have one to three layers. Extra hidden layers make these networks deep and extends their predicitive capability as higher order features can be captured in the larger hierarchical structure these deep networks represent. However, adaptions to these networks can provide them with different capabilities. When, instead of feed forward networks in which information is only propagated from one layer to the next, information is also led back to previous layers the networks are called recurrent neural networks (RNNs). First, an introduction to these RNNs will be given, after which LSTMs are introduced, which have proven to be some of the best performing algorithms in past research.

Recurrent neural networks

RNNs are similar to feed forward neural nets, but have 'internal memory' by means of hidden states and feedback loops [46]. A hidden state vector is passed from one node to the other containing information on previous inputs, calculations and outputs, essentially causing that the same input to a node can lead to a different output if the past inputs to the preciding series of nodes is different. Often in RNNs every layer represents a time step or step in a sequence. This way, time series data is better represented, as information in for example two time steps prior to the current step can be very relevant for the next

step. However, as a weight is applied on every step, the historical information vanishes quickly over time. This problem is called 'gradient vanishing', leading to short term memory.

This problem is solved by using Long Short-Term Memory neural networks [26], an adaption of RNNs which are better at 'remembering' past data on the long term, solving the problem of gradient vanishing. It does so by propagating the previous state through each cell and only adapting this state where necessary instead of applying a weight on the cell state as a whole. This is done by a series of operations which are grouped in 'gates', which are defined as follows. An overview of a LSTM cell can be found in fig. 2.3.

- **Forget gate:** This part of the cell determines which details of the previous state should be discarded, and is represented by the first sigmoid function on the left of the figure.
- **Input gate:** Determines which information from the inputs should be passed to the 'memory' and which weight should be assigned to this information. The second sigmoid function makes the decision on which information to pass, while the tanh function assigns a weight to this information.
- **Output gate:** This gate provides an output from the system and determines on which information the next cell should base their decisions. It does so by means of a sigmoid function, the right-most sigmoid function in the figure, to determine the information from the previous cell and the inputs. The tanh function again determines the weight assigned to this information.
- **Cell state:** The cell state is represented by the black arrow on top of the cell. It propagates information from all previous cells through the network, and at every cell or timestep this information is adapted by the forget gate if the information is not deemed relevant, and new information is added by the input gate.

It can be noted that each gate is essentialy a small neural network on its own. The structure of the network, with cells in sequence able to remember past information, makes it a suitable algorithm to perform time series prediction with. Also, the tanh function regulates the output of the LSTM. Many variations on the LSTM neural network exist, allowing for example multivariate inputs or multi-step outputs. Seeing as the problem of 4D trajectory prediction requires a multitude of variables as input and multiple steps ahead as prediction, this is the type of network that has to be implemented for the predictor.

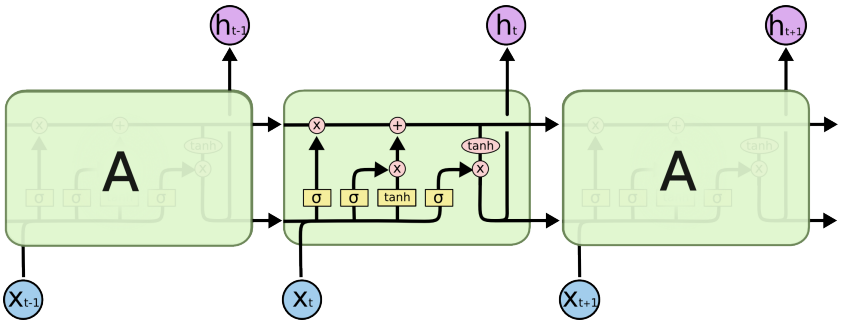


Figure 2.3: Structure of a LSTM neural network, detailing the state vector which is propagated and on top and the inputs and previous states which are used to adapt the state vector via various operations on the bottom. Source: [2]

2.2.4 Dimensionality reduction

Before applying a clustering method, it can be beneficial to perform dimensionality reduction to the input data of the clustering method. First of all, data for machine learning has to be processed in such

[2] www.tensorflow.com

a way that it can be handled by the algorithms. This will be further explained in chapter 3, but after this pre-processing a number of dimensions will remain. Every dimension or feature in this data can be indicative for the predictor, but having too much data and features available increases the chances of for example overfitting. Therefore, reducing the amount of features or dimensions present in the data increases the predictive performance of the models, and increases the computational efficiency. Two main forms of dimensionality reduction exist:

- **Feature selection:** the most relevant subset of features from the original set of features is selected, reducing the size of the set.
- **Feature extraction:** from the original set of features the most relevant information is derived, which is then used to construct a new set of features.

One of the state-of-the-art methods is principal component analysis (PCA), which is a feature extraction method. Its goal is to capture the most relevant information and map this onto a feature space which is lower in dimension than the original one. Principle components stand for orthogonal axes, which in the case of PCA should point in the direction of maximum variance, of which a simple example can be found in fig. 2.4. It identifies patterns in data based on the correlation between features by means of constructing a covariance matrix, of which the eigenvectors and eigenvalues are then used to rank the features. The top ranking features are then used to construct a projection matrix, which is used to transform the input data to a new feature space. Using PCA, clustering can be faster and more effective and this is introduced in the next section.

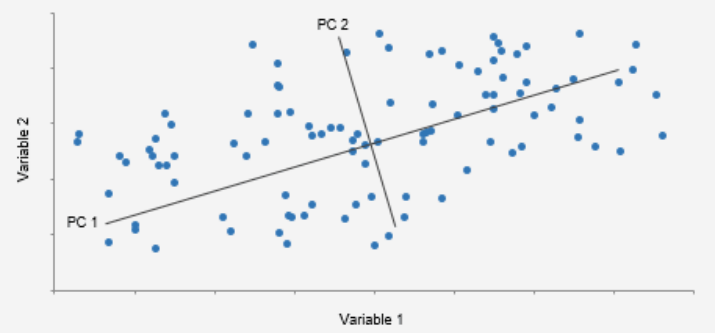


Figure 2.4: Principle Component Analysis (PCA) tries to map the original features of a dataset onto a new set of features, indicated by the orthogonal axes PC1 and PC2. Source: [3]

2.2.5 Clustering

After dimensionality reduction the trajectories can be clustered. As indicated in section 2.1, the trajectories will be clustered using Density Based Spatial Clustering for Applications with Noise as it outperformed comparable algorithms for aircraft trajectories. It was developed by Ester el al. [17] and is based on two parameters:

- **Eucledian Distance, ϵ :** this parameter determines how close data entries should be to each other to be defined as part of a cluster. The distance is measured in terms of the Eucledian Distance. If ϵ is chosen too small, most data will defined as outliers, while if it is too large, the clusters will become very large and will overlap. A K-distance graph can help in determining the value for ϵ.

[3] https://www.statistixl.com/features/principal-components/

- **MinPoints, μ :** this parameter defines the minimum number of data entries to call something a cluster or dense region. The value for μ should be at least larger than the number of dimensions in the dataset. Also, the larger the dataset, the larger the value for μ should be.

The algorithm defines each data entry as a 'core point', 'border point' or a 'noise point'. Core points are those that have at least μ data entries within their ϵ perimeter. Border points are those that don't have that many data entries within their ϵ perimeter, but are within ϵ distance of a core point. The core and border points together form clusters. Noise points are those that don't have either μ points or a core point in their ϵ perimeter and are therefore defined as outliers.

The latter provides the main advantage of this algorithm, which is its robustness against outliers. Also, the number of clusters present in the data is determined by the algorithm, in contrary with other common clustering methods such as k-means clustering. Furthermore, DBSCAN is able to find clusters of arbitrary shape. Also, the clustering method is relatively stable in comparison with k-means clustering, in which each data entry of a cluster has influence on it's centroid and therefore on the shape of the entire cluster. Disadvantages are that the parameters of the algorithm are not optimised through a set number of steps and that performance can deteriorate if the clusters have a non-uniform density.

With all algorithms for the predictors defined and the techniques behind them explained, the prerequesits of the predictions, namely the input data, can be investigated. This will be presented in the next section.

2.3 Input data

In this section, the inputs and prerequisites for trajectory prediction will be presented. These consist of several data sources that together form a basis to learn a machine learning model on. First, an analysis will be performed of literature in which the influence of several types of data on the quality of a trajectory predictor are described in section 2.3.1. Then, these types of data are further described and their sources chosen in subsections 2.3.2 to 2.3.5, along with an analysis of the data used in previous trajectory prediction research. Finally, the amount of data required is analysed in section 2.3.6.

2.3.1 Background information

In order to predict trajectories based on historical, flown trajectories, data is required that is an accurate representation of the trajectory. However, the reasons why the flight was executed as described in this trajectory is also important for future prediction. Understanding why an aircraft has a delay, was redirected such that the flight path was elongated or was able to 'cut corners' by means of a direct-to from ATC are of vital importance in predicting future trajectories. Without these supporting facts and figures all predicted flights will roughly look the same, resulting in errors in the prediction.

First of all, several assessments of the influence of input data on trajectory predictors have been performed in literature. Mondoloni et al. [51] [38] have shown the importance of turn dynamics, aircraft weight, aircraft intent, aircraft performance and interim level-flight sections for trajectory prediction using APMs. Similarly, Rudnyk et al. [45] have shown that wind, vertical speed intent and CAS/mach speed settings are the main sources of errors in trajectory prediction and also analysed the effects of the bank angle, air temperatures and interim level-flight sections on trajectory prediction using APMs. Weitz et al. [62] have shown that in general, vertical errors originate from errors in aircraft weight, horizontal errors are due to wind conditions and flight duration errors are due to the true airspeed, which is again linked to amongst others the wind conditions.

From these sources, it can be concluded that for trajectory prediction using APMs an accurate representation of the aircraft weight, meteorological conditions, speed settings, aircraft performance and

aircraft intent or flight plans is necessary. When performing trajectory prediction without using APMs, speed settings and aircraft performance parameters become less important, as well as aircraft intent. This is due to the fact that the machine learning models should provide a representation of the to be flown flight path instead of this being an input to the predictor. Also, seeing as aircraft weight is an input for APMs but are no direct input to other types of prediction, the aircraft weight can also be represented and approximated by other means of information. Data indicative of the aircraft weight can be for example the type of aircraft, the operator, origin or destination, day of the week and the percentage of the trajectory already executed. The factors affecting the aircraft weight can be used in the prediction without having to estimate a number for each aircraft. Seeing as EHAM is an airport with six runways at different angles, the ETA and final part of the trajectory will probably be dependent on the runway availability. Therefore, the influence of the runway availability on the quality of the predictor will be assessed in this study as well.

The representation of the flown trajectories will be presented in section 2.3.2, after which the route data containing the origin or destination of a flight will be detailed in section 2.3.3. Then, the meteorological data will be further explored in section 2.3.4. The runway availability will be investigated in section 2.3.5. Finally, it is of importance to know the the amount of historical data necessary to make reliable predictions. This will be presented in section 2.3.6.

2.3.2 ADS-B data

The most important input to the to be built trajectory predictor are the historical flight trajectories. These form a basis for the machine learning models to learn on and form the reference with which predicted trajectories are evaluated. These historical trajectories are generated based on aircraft position data, which can be harvested by means of radar data or using data which is being broadcast by all commercial aircraft by means of Automatic Dependent Surveillance - Broadcast (ADS-B) data. All sources from section 2.1 are using (a processed form of) ADS-B or radar data in their research. ADS-B data is downlinked by aircraft through Mode-S Extended Squitter (1090 MHz) and is openly receivable by everyone with an antenna [53]. Also, ADS-B data is accessible through many online sources which gather the data from contributors all around the world[4]. The data is constituted of in total 112 bits, which together transfer the following information: aircraft identifier (ICAO address), aircraft position (lat/lon, altitude) and aircraft velocity (heading, ground speed and roc). The decoding and processing of ADS-B data is covered in section 3.3. For this research, ADS-B data received by an antenna at the faculty of Aerospace Engineering at TU Delft will be used, which has a coverage extending well over the Dutch FIR, see fig. 2.5, and is openly accessible. Due to the location of this receiver only partial trajectories with Amsterdam Schiphol Airport (EHAM) as destination will be used in this research.

From the aircraft position data flown trajectories can be extracted which form the basis of the direct trajectory predictor. Then, the aircraft velocity data is used as supporting data for the predictor and can be used to check the accuracy of a predictor. Some research uses Enhanced Mode-S data, which can also include additional information on roll angles, true airspeed, mach numbers, magnetic headings and other more advanced types of aircraft state data [53]. However, as this data is not openly accessible it is not considered as input to this study.

2.3.3 Aircraft route data

Other important indicators of a trajectory are the origin or destination (O/D) of a flight, which operator is executing the flight and which aircraft type executes the flight. When it is for example known where a flight originates from, a typical trajectory can be assigned to such a flight. In order to assign an O/D or operator to trajectories, route data is necessary. However, seeing as data on which operator flies which aircraft to which destination is commercially sensitive information, no open data source was

[4]www.flightradar24.com, www.flightaware.com, amongst other sources

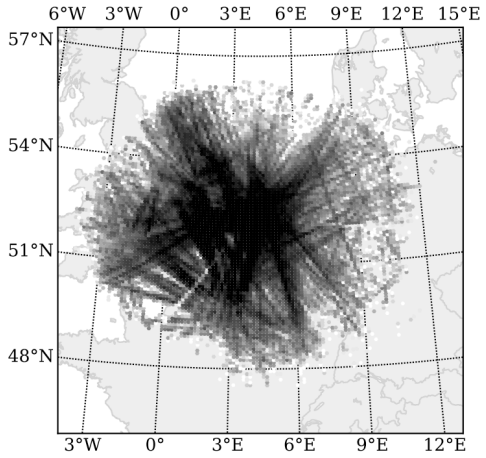


Figure 2.5: Coverage of the ADS-B antenna at TU Delft. The different shades of grey indicate the ADS-B data density. Common flight routes are distinguished by darker shades of grey. Image from Sun et al. [52]

available. The website of Schiphol Airport does publish all flights arriving and departing at the airport daily. Along with the origin and destination, the operator, aircraft type, planned and executed time of arrival / departure, the callsign and the aircraft registration are posted. It is chosen to to harvest this data daily using a web scraping algorithm. Seeing as EHAM is the only airport considered in this study and this source contains all required information, it is deemed sufficient for this purpose. Also, it provides us with additional information which might be indicative of a trajectory, namely the delay of the flights.

2.3.4 Meteorological Data

As stated in section 2.3.1 meteorological data is of importance to the accuracy of trajectory predictors. The most important factor are the wind conditions, after with the air temperature has the most influence. These wind conditions consist of both the magnitude and direction of the wind in a 3D frame. For the purpose of aircraft trajectory prediction a data source is needed which has coverage over a large area and up to high altitudes. In previous research, several data sources have been used.

Data from the National Oceanic and Atmospheric Administration (NOAA) of the US government, specifically 'Global Forecast System' data[5] is the most commonly used source for meteorological data in the aircraft trajectory prediction literature assessed in section 2.1. It provides global coverage on a 0.5 x 0.5 deg grid and is updated every 3 hours. It is used by amongst others Hernández et al. [25], de Leege et al. [15] and in the DART research [18]. Other frequently used sources from the NOAA are 'Rapid Update Cycle', now replaced by 'Rapid Refresh'. These are amongst others used by Ayhan et al. [7], [6] and Lymperopoulos et al. [37]. However, these sources only cover North-America and are therefore not suitable for this research. The meteorological dataset with the highest gloabl spatial resolution currently available is ECMWF ERA-5 [6]. It is the successor of ECMWF ERA-interim which has been used by for example Zhang et al. [63]. The ERA-5 data has a spatial resolution of 0.25 x 0.25 deg over 137 levels of altitude and provides an hourly update of the meteorological conditions. This being the source with the highest time- and spatial resolution, it is chosen as the source of meteorological data to use in this research.

[5]https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs
[6]https://www.ecmwf.int/en/forecasts/datasets/reanalysis-datasets/era5

2.3.5 Runway usage Amsterdam Schiphol Airport

Most research in literature performs trajectory prediction on specific routes. However, for this research the goal is to predict a trajectory for all incoming flights at EHAM. With six runways and strict rules on which runway can be open under which meteorological circumstances, while also taking into account the impact on the surrounding neighbourhoods it is not trivial which runways are in use for landing at time of arrival for predicting the trajectory. Especially for the last part of the flight it is expected that this has a major influence as aircraft will have to approach the airport from another direction if the usage of the runways flip, leading to large errors both in spatial terms as in timing. The runway availability is regulated by the Dutch ATC, the LVNL, and is posted online for each time slot of five minutes [7]. This data can be used for training the machine learning models. When predicting new flights, a prediction of the runway availability has to be available. Commercial solutions for this purpose exist, however this is out of the scope of this research. As the predictor will be tested using historical data in this research, the true availability is available. Although this will be slightly more accurate than the predicted availability, the difference in these will not be large as the runway availability does not change often during a day. The latter can be concluded from analysing the harvested data.

2.3.6 Amount of historical data required

In order to be able to accurately predict trajectories the data set containing the historical trajectories has to be large enough. To determine the required size of the data set, former research is analysed. It can be concluded that two types of research exist. First of all, there are predictors which are trained on a specific part of a route, such as one approach route or only one climb phase section, or which are trained on a specific aircraft at a specific airport. These predictors are typically trained with 500 to 10,000 historical trajectories [22], [7], [6], [15], [57], [35] and [25]. On the other hand, predictors that predict trajectories for larger parts of airspace and for multiple or all aircraft types they come across use far larger data sets, often consisting of 40,000 to 100,000 trajectories [30], [31], [29], [59], [58]. Considering the size of the airspace in which the trajectories will be predicted, the Dutch FIR, and if no limits will be set for the routes and aircraft types for which trajectories will be predicted, a large data set of at least 40,000 historical trajectories will be required for this research.

2.4 Analysing trajectory predictions

When performing trajectory prediction, several approaches can be taken with different goals in mind. Some research only use it for the prediction of delay, in order to ensure efficient handling at the destination airport. Others use trajectory prediction for climb prediction, or to predict the descent length. With these approaches, different measures of performance are used, such as time-only, vertical cross-track deviation or along-track deviations.

Ryan et al. [47], Mondoloni et al. [39] and Paglione et al. [41] identified frameworks for assessing the quality of a trajectory predictor, of which the most important factor is the accuracy of the prediction. The metrics to assess accuracy will be detailed in section 2.4.1. Another important factor is the speed of a trajectory predictor, which is detailed in section 2.4.2. Finally, the stability of a predictor is considered to be important for the purpose of ATC and is described in section 2.4.3. In section 2.4.4 the quality assessment chosen for this research will be presented.

[7] https://www.lvnl.nl/omgeving/baangebruik

2.4.1 Metrics of accuracy

Two main categories of errors in 4D trajectories can be identified, spatial and time errors [47]. Spatial trajectory errors are the offset in location at a certain point in time, whereas time errors are the offset in time at a specific point in space. Time errors are often defined at a key location in a trajectory, such as the top-of-descent or landing: ETA prediction, which is done by de Leege et al. [15], Wang et al. [59] and Hernandéz et al. [25].

Spatial trajectory errors can be measured at any point in time. The horizontal error consists of the along-track and cross-track error and is basically a vector decomposition. Along-track errors are the difference in location found along the trajectory, while cross-track errors are the errors found perpendicular to the trajectory. Tastambekov et al. [57] and Hernandéz et al. [25] use such accuracy measures. Most research, however, focuses on only the altitude component of the spatial errors, often by predicting the flight level during the climb phase of the trajectory, such as Alligier et al. [2], [3], [4], Hadjaz et al. [22], as well as Tastambekov et al. [57] and Hernandéz et al. [25].

Another metric used in trajectory prediction is the error in velocity at a certain point in space or a certain point in time. Hrastovec et al. [30], [31] as well as Lin et al. [35] determine the difference in lateral speed and ROC in order to assess the quality of the predictor. This method is useful when analysing aircraft performances and for the very first and last flight phases. However, when using the prediction to support ATC in their decision making processes, this metric is not as much of use as the spatial and time errors. Also, errors in velocity result in errors in spatial and time metrics, and are therefore already captures by these metrics.

Finally, the total (ground-) distance covered can also be used as a metric. Alligier et al. [5] use the ground distance of the descent length from cruise to a set final altitude. This method is helpful in predicting when an aircraft will start its landing procedures but does not indicate the trajectory it will follow and the velocities it will have in order to determine at which time an aircraft is where.

2.4.2 Speed of the predictor

The speed of the predictor is an important indicator of performance for the predictor. A predictor which provides very accurate results but requires too much processing power will result in a useless tool. The processing power can be measured at two steps in the prediction process. First, when the prediction model is built, it should be noted how much time the model takes to learn from historical data. Then, when new data is fed to the model, it should be determined how much time it takes to predict a new trajectory. These two metrics determine the speed of the predictor and are of importance for the implementation of any tool resulting from this type of research.

2.4.3 Stability of predictions

As introduced by Mondoloni et al. [39], the stability of a trajectory prediction is determined by the changes in the predicted trajectory over time. Although this metric has been proposed in several frameworks for assessing the quality of predictors, no recent literature assessed the stability. It is of importance as a prediction which changes significantly over time is not of use to ATC. Part of this is captured by the accuracy metrics, but if a prediction constantly changes within the limits of this accuracy, it can still deteriorate the usefulness for decision making. It can be measured by considering a prediction to be the the baseline and determining the cross track errors of a new prediction with respect to this baseline. This, however, would be a computational burden as for each update cycle all these along and cross track errors would have to be determined. More efficient would be to determine the change in ETA between predictions as this can change the sequence of arriving aircraft, which information is of direct use to ATC.

2.4.4 Chosen quality assessment

In this study the goal is to perform research that will in the end provide enhanced information to ATCo, to support them in their decision making. For this purpose first of all ETA prediction is important. Whether the ETA is determined at landing or at a certain point in a trajectory, such as for example a limiting sector in the airspace, it is important to know for an ATCo which aircraft will be in his sector at which time, ahead of time. This way, the ATCo can ensure safe and efficient guidance without too many deviations to solve near-future problems encountered only a few minutes before they occur. Together with ETA prediction, the route flown by an aircraft is important to ATC. Therefore it should be known what the spatial error is. Therefore, both the horizontal as the vertical error will be measured over the whole trajectory.

Next to accuracy measures, the speed of the predictor will be determined using the two methods described in section 2.4.2: during the learning/training and during the prediction/testing phase of the predictor. Also, the stability of the predictor will be analysed by the change in ETA between predictions. This analysis was not found in research and is therefore assumed not to have been performed before in aircraft trajectory prediction.

In this study only deterministic trajectory prediction is considered. This form of trajectory prediction does not take uncertainties in the prediction into account by showing multiple possible trajectories. Only the most accurate prediction is used and this prediction is assumed to be true. For use cases such as conflict detection, multiple possible predictions can be made in order to detect possible conflicts and anticipate on these. However, in this study only the most probable prediction is needed and will therefore be the only one taken into account in the quality assessment.

3

Data Processing

The data presented in chapter 2 will be used to perform the trajectory prediction with. In order to use this data for the baseline simulations or for the to be developed machine learning predictors, the data has to be in a format these algorithms and systems can handle. Therefore, it is investigated in which form the data is received, what the desired format of each data type is and how this can be achieved. Also, this chapter presents how the data from various sources can be combined in order to build enriched data sets on which the predictors can train. First, the required data format for the machine learning algorithms will be presented in section 3.2. The handling of the core of the data used in this study, ADS-B data, will be detailed in section 3.3, after which the meteorological data will be dealt with in section 3.4. Then, the route data will be presented in section 3.5. Finally, the runway availability will be presented in section 3.6, after which the coupling of the various data sources will be detailed in section 3.7. An overview of the outlier detection and removal will be given in section 3.8. Concluding, the resulting data structure is shown in section 3.9 and an analysis and overview of the resulting data in section 3.10.

3.1 Required formatting for BlueSky simulations

The baseline simulation using APPs will be executed in BlueSky and it should be investigated how the input data for these simulations should be formatted. BlueSky requires an aircraft to be defined based on the aircraft type and current position, speed and heading. Then, waypoints can be added along with a desired altitude and speed, which are then used to simulate the trajectory. Therefore, for each trajectory the parameters and their format as stated in table 3.1 are required.

In order to obtain these parameters, several steps have to be taken. First of all, the initial conditions such as aircraft type and position can be extracted from the available ADS-B and route data. The required waypoints, however, are not readily available. In order to obtain these, it is chosen to assign waypoints based on the executed, true trajectory. Based on the location of the incoming flight, a STAR is assigned. The planned waypoints for an aircraft originating from the north-west of EHAM would then be TOPPA, SUGOL and SPL and from the east for example NORKU, ROBIS, ARTIP, SPL. Also, based on the STAR, a target FL and SPD is provided at either SUGOL, RIVER or ARTIP. Seeing as no information is available on the runway used for landing in such a system, SPL is used as the terminal waypoint for each trajectory, with FL 10 and a speed of 150 kts. This method is representative of the flight plans used currently by ATC to predict trajectories and is therefore used to compare with the methods introduced in this study.

Table 3.1: Parameters required for the baseline simulations

Parameter	Unit	Example
ACID	-	TID
Aircraft type	-	A320
LAT	deg	54.964743
LON	deg	2.871752
ALT	FL	FL250
SPD	kts	300
HDG	deg	170
WPT # 1	-	TOPPA
ALT # 1	FL	FL150
SPD # 1	kts	220
. . .	. . .	. . .
WPT # x	-	EHAM
ALT # x	FL	FL10
SPD # x	kts	150

3.2 Required data formatting for machine learning algorithms

In section 2.1, several machine learning algorithms have been introduced. In this study, these algorithms will be used to perform calculations on input data in order to provide predictions of future time steps. These calculations will have to be performed on numerical data, however a lot of text-based parameters are present in the input data. Therefore, these parameters have to be transformed into numerical data, which can be performed using One-Hot Encoding. This will be presented in section 3.2.1.

The same calculations performed on numerical data are sensitive to the size of a value. If, for example, the average value for the ground speed is around 200 kts while the average value for longitude is 2 degrees, the algorithms tend to rely more on the value for the ground speed rather than on the value for longitude, simply because its value is higher. This behaviour is suppressed by means of scaling and standardising the numerical values of the parameters, which is further detailed in section 3.2.2.

3.2.1 One-Hot Encoding

Text-based data present in the input data essentially is categorical data. The 'Airline' of a trajectory can be seen as a categorical parameter with hundreds of categories and the same holds for the 'Origin', which is a category consisting of all airports Schiphol is connected to. In order to transform this categorical data into numerical data, One-Hot Encoding can be used [24]. This method converts categorical data into numerical data by assigning a new parameter to each category while these new parameters can only have a value of 0 or 1 (binary). It is called One-Hot Encoding as only one value in a vector is 'hot' or 'true' after encoding.

An example of One-Hot Encoding is the category 'gender', which can, for example, have three values: 'female', 'male' or 'other'. If a subject is male, in One-Hot Encoding this translates to the following vector: [0, 1, 0]. A common mistake is to categorise by only translating the categories into numbers, such that : ['female', 'male', 'other'] = [0, 1, 2]. This would imply that 'other' is better or higher than 'male' as its value is higher, while it is simply another category. Finally, decision trees, which are the basis for GBM, are able to handle categorical data as they can split the branches based on this data. However, they can also handle data sets which are encoded, so the same format will be used for all

algorithms. A disadvantage of One-Hot Encoding is when new categories, such as for example a new origin or new airline is found in the test data. In such a case, no category can be assigned and therefore no prediction can be performed. This should be mitigated by ensuring the training set is large enough to capture all categories present in the data.

3.2.2 Scaling and Standardising

As explained, numerical data has to be scaled and standardised before using it to train the machine learning models on. Seeing as the algorithms expect standard normally distributed data, which is a Gaussian distribution with zero mean and unit variance, a standard score is calculated for each value of each parameter. This calculation is presented in eq. (3.1) and relies on μ, the mean of the series of values for each parameter and σ, the standard deviation of this same series. This scaling is performed for each parameter separately, where μ and σ are based on the training set only. The test set is then transformed to this standard score using the same values for μ and σ.

$$z = \frac{x - \mu}{\sigma} \tag{3.1}$$

3.3 ADS-B data processing

As introduced in section 2.3.2, the ADS-B data used for this study is received by an antenna at the faculty of Aerospace Engineering at TU Delft. Here, the raw ADS-B data is stored and made available to students and researchers. This raw data consists of messages of 112 bits which contain information on the position and velocities of the aircraft [53]. Using software developed by Sun et al. [52] this raw data is decoded to a format consisting of the parameters as shown in table 3.2. The callsign is often not broadcast and seeing as it encompasses the same information as the ICAO ID for this research, it is omitted and not taken into account for the remainder of this research.

Table 3.2: Parameters available in decoded ADS-B data

Parameter	**Unit**
Timestamp	-
ICAO ID	-
Latitude	deg
Longitude	deg
Altitude	ft
Ground speed	kts
Rate of Climb	ft/min
Track	deg
Callsign	-

A series of subsequent ADS-B messages from the same aircraft can in turn be converted to a trajectory. In order to determine which ADS-B message belongs to which trajectory, Sun et al. [54] also developed software which is able to construct these trajectories. It uses DBSCAN to cluster points into trajectories based on both location and position as well as time. The latter is of importance to split the trajectories of an aircraft which executes several flights during a day in the Dutch FIR from each other. Using this software trajectories are reconstructed which run from the range of the ADS-B receiver up to an airport in the Netherlands as well as to the border of the range of the antenna again.

In order to provide regular trajectory prediction updates, these trajectories need to consist of densely

and regularly spaced points. The data received by the antenna is irregularly spaced, however, which has to do with the changing velocity which the aircraft have with respect to the antenna location, the settings of the transmitter of the aircraft, as well as with the distance of the aircraft to the antenna. The latter causes some messages to be discarded as they are incomplete or have errors due to the long distance travelled and can therefore contain noise. The former two reasons cause slight offsets in the spacing of the messages. In order to solve this problem, the data is interpolated after extracting the trajectories. Seeing as ADS-B messages are broadcast with an interval of less than a second, the behaviour of the aircraft is assumed to be linear for these small timesteps. Therefore, linear interpolation is used.

3.4 Meteorological data processing

The meteorological data sourced from the ECMWF, ERA-5, consists of files containing the temperature and u, v, w velocities for 37 altitudes on a 0.25 x 0.25 deg grid over the whole earth. The altitudes at which this data is available can be found in table 3.3. The altitudes specified in this source is the barometric altitude expressed in hPa. Seeing as 1 hPa = 1 P_{mb}, or millibar, this can be translated into an altitude in feet which is compatible with the ADS-B position data. The equation for this translation is sourced from the NOAA [1] and is presented in eq. (3.2) and the resulting altitudes in table 3.3.

$$h_{ft} = \left(1 - \left(\frac{P_{mb}}{1013.25}\right)^{0.190284}\right) \cdot 145366.45 \tag{3.2}$$

Table 3.3: Pressure altitudes at which the meteorological data is available, including conversion to feet

$h_{baro,hPa}$	h_{ft}	$h_{baro,hPa}$	h_{ft}
1	106415	400	23564
2	100923	450	20804
3	97359	500	18281
5	92458	550	15955
7	88959	600	13795
10	84998	650	11776
20	76487	700	9878
30	70962	750	8088
50	63367	775	7229
70	57945	800	6392
100	51806	825	5576
125	47748	850	4779
150	44302	875	4002
175	41293	900	3242
200	38615	925	2499
225	36195	950	1772
250	33985	975	1061
300	30053	1000	364
350	26620	-	-

It can be noted that these data sets cover a large range of altitudes than required for our models. First of all, the aircraft considered will in general not fly over 40,000 ft. Therefore, it is chosen to only extract the data between 150 and 1000 hPa. Furthermore, the data sets cover the entire globe. After an analysis

[1] https://www.weather.gov/media/epz/wxcalc/pressureAltitude.pdf

of the origin and positions of the ADS-B messages it is chosen to extract data on longitudes 357.00 up to 12.75 and latitudes from 47.00 up to 56.75. Both in altitude as in horizontal spacing a margin has been taken into account in order to ensure all trajectories can be fitted with meteorological data. The meteorological parameters available can be found in table 3.4. All values for these parameters are scaled and standardised using the methods described in section 3.2.2.

Table 3.4: Meteorological parameters available with units from ERA-5

Parameter	Unit
Temperature	K
u wind velocity	m/s
v wind velocity	m/s
w wind velocity	Pa/s

3.5 Route data processing

As stated in section 2.3.3, aircraft route data is not openly available. Therefore, it is chosen to harvest this data from the website of Amsterdam Schiphol Airport[2], by means of webscraping. Every flight arriving or departing at the airport is published on the website, along with additional information and parameters of the flight. These parameters can be found in table 3.5. It is chosen to first harvest this data as it is only available for one day, and then investigate which parameters enhance the predictive capabilities of the models by performing tests with and without these parameters. Some parameters are interchangeable and/or are expected to provide better results than other parameters. An example of this would be that it is expected that the Aircraft Type provides better predictive capabilities than Aircraft Manufacturer and seeing as the latter is indirectly included in the Aircraft Type it makes the former parameter redundant. Therefore, it is chosen not to use the Aircraft Manufacturer parameter for the remainder of this study.

The planned arrival time will, along with the true arrival time and timing parameters, be used to train the prediction models. However, the latter two will be used for validation, while the first one will be an input parameter for the predictors. The true arrival time can then be compared with the ETA provided by the prediction model. It can be investigated whether the ETA predicted in the previous time step should serve as an input to the prediction of the next time step. This way, it can be investigated whether delay has an influence on the behaviour of the pilots and ATC, such as an extra shortcut or direct-to or higher velocities.

3.6 Runway usage EHAM processing

The availability of runways at Amsterdam Schiphol Airport is published on the website of the Dutch ATC[3] on a 5-minute time scale. It is harvested by means of webscraping and an example of the resulting data can be found in table 3.6. This parameter has not been tested before in literature and therefore its predictive capabilities should be investigated thoroughly. Seeing as each airport has a different layout the predictive value of this parameter can not be generalized to any airspace. Also, seeing as the LVNL schedules the runway availability not only based on the time of the day and the wind conditions but also based on rules limiting the nuisance for the surrounding areas, the results for this part of the research may be very specific to Schiphol Airport. It is expected that at most airports the wind conditions and time of the day provide a fairly good approximation of the runway availability and both of these parameters are usually already taken into account in the predictors.

[2]https://www.schiphol.nl/en/arrivals/
[3]https://www.lvnl.nl/omgeving/baangebruik

Table 3.5: Parameters harvested from the website of Amsterdam Schiphol Airport

Parameter	Example
Year	2019
Month	11
Day	11
Airline	KLM
Aircraft Registration ID	PHBHD
Aircraft Manufacturer	Boeing
Aircraft Type	787-9 Dreamliner
Origin	Mumbai
Flight Number	KL878
Planned Arrival Time	07:25
True Arrival Time	08:35
Timing	Delayed

Table 3.6: Layout of the runway availability data harvested from the LVNL website

Time	Kaag	Buitenveldert	Oost	Aalsmeer	Zwanenburg	Polderbaan
09:35	take-off	closed	closed	closed	closed	landing
09:40	take-off	closed	closed	closed	closed	landing
09:45	take-off	closed	closed	closed	closed	landing
09:50	take-off	closed	closed	closed	closed	landing
09:55	take-off	closed	closed	closed	closed	landing
10:00	take-off	closed	closed	closed	closed	landing
10:05	take-off	closed	closed	closed	closed	landing
10:10	take-off	landing	closed	closed	closed	closed
10:15	take-off	landing	closed	closed	closed	closed
10:20	take-off	landing	closed	closed	closed	closed
10:25	take-off	landing	closed	closed	closed	closed
10:30	take-off	landing	closed	closed	closed	closed
10:35	take-off	landing	closed	closed	closed	closed
10:40	take-off	landing	closed	closed	closed	closed
10:45	take-off	landing	closed	closed	closed	closed
10:50	take-off	landing	closed	closed	closed	closed
10:55	take-off	landing	closed	closed	closed	closed
11:00	take-off	landing	closed	closed	closed	closed

3.7 Coupling of data sources

With the required data format presented and the input sources defined, the last step is to enrich the ADS-B data, which essentially is the trajectory data, with the additional data presented in the previous sections. In order to couple the various sources, commonalities have to be found between the data sets. Enriching the ADS-B data is presented on a per-type basis and is not per se sequential: meteorological data can be added while not adding route data and vice-versa in order to test the predictive value of each source. The combinations will be presented in chapter 4.

3.7.1 ADS-B and meteorological data

In order to enrich the ADS-B data with meteorological data, several assumptions have to be made. The meteorological data is rather sparse and therefore it is chosen to interpolate this for each individual trajectory point. It is assumed that the parameters have a linear behaviour between the available points, so the parameters are interpolated linearly. The interpolation is performed both in time as in 3D space, so $2^4 = 16$ data points are considered for each parameter of each time step and weights are assigned according to the proximity of the aircraft to each available data point.

3.7.2 ADS-B and route data

In order to couple the ADS-B to the route data several steps have to be taken. First of all, the ADS-B data contains the ICAO address of the aircraft, whereas the route data contains the registration ID (REGID) of the aircraft. To couple these, the World Aircraft DB of J. Sun [4] is used, which contains both the ICAO address as the REGID for nearly all aircraft. The entries for aircraft missing in this dataset were extracted from Flightradar24[5] manually. This allows for filtering the available trajectories to only the trajectories of aircraft visiting EHAM. However, most aircraft execute multiple flights a day to and from EHAM or other airports which are in range of the ADS-B antenna. Therefore, the database will contain multiple possible partners for each route entry. In order to assign the correct trajectory to each route, a common feature should be found on which to base such a decision on. In this case, it is unknown whether a trajectory represents a departing or an arriving flight (although this is deducible by looking at the data). However, it is known at which altitude, lat/lon location and time the trajectory ends. If this altitude is low and the lat/lon location corresponds roughly to the location of EHAM, it can be said that an aircraft is arriving. Then, if the timestamp of the last time step of the trajectory corresponds to the landing time of the route of the same REGID, a trajectory can be coupled to this route. A problem arises when solving this problem, which is that the data available is slightly noisy. Due to objects blocking the ADS-B messages from aircraft close to the ground aircraft, the time delta between the true landing time and the last timestamp from the trajectory can become large. Therefore, a relatively large maximum time delta of 20 minutes is chosen. All values chosen as criteria for this coupling procedure are defined as follows:

- **LAT range:** [52.219, 52.441]
- **LON range:** [4.533, 4.971]
- **Altitude:** Max: 3000 ft
- **Time delta:** Max: 20 minutes

3.7.3 ADS-B and runway data

As a first test, the runway availability at the planned arrival time from the route data will be incorporated in the enriched data sets. This does mean, however, that the runway availability can only be added to the ADS-B data when it is already enriched with route data. Furthermore, seeing as some aircraft will arrive earlier or later than planned, in a later stage the runway availability at the ETA can be used to improve the predictive performance. This would also eliminate the dependency on the route data for coupling, however, it would result in an iterative process as the runway availability will have an influence on the ETA and vice versa.

[4] https://junzis.com/adb/, accessed on 20-10-2019
[5] https://www.flightradar24.com, accessed on 01-11-2019

3.8 Outlier detection and removal

When working with real-life data sets noise is inherently present and this has to be accounted for. An example of noise in our data sets could be incorrect data points in the trajectories from the ADS-B data, originating from erroneous decoding and interference of the ADS-B signal through the air. On a higher level, noise could be a very a-typical flight due to unconventional circumstances which causes an entire flight to be an outlier.

In the various steps from the raw data towards the final data set outlier detection and removal is already present. The raw ADS-B data is formed into trajectories by means of DBSCAN, which clusters the data points belonging to the same trajectory. DBSCAN performs outlier detection and removal on these single data points and is also applied on the trajectories as a whole, eliminating trajectories which are outliers. Also, by performing the coupling between the routes and the trajectories using the four delimiters (LAT range, LON range, maximum altitude and maximum time delta) outliers in both the route data set and trajectory data set are removed. Seeing as the ERA-5 meteorological data set is sourced from the ECMWF and has been re-analysed before publishing, it is considered to be accurate and without significant noise. Finally, when applying DBSCAN for clustering during the experiments, additional outliers are removed.

3.9 Data processing overview and structure

This section presents an overview of the data processing and combining. Then, the resulting data structure will be presented. The data sources presented in the previous sections are to be combined into a data set on which the machine learning algorithms can learn. The flow from input data to these data sets is visualised in fig. 3.1.

The parameters available in the final data sets can be divided into parameters which differ for every single point in a trajectory and parameters which contain information on the trajectory as a whole. For computation and storage efficiency, these data sets are separated, so as not to store information on a trajectory as a whole for every single data point. The parameters available on every single data entry are shown in table 3.7, and the parameters on each trajectory as a whole in table 3.8. This represents all parameters available in this study. How these parameters will be used will be further explained in chapter 4.

Table 3.7: Data available at every data point of the trajectory

Parameter	**Unit**	**Source**
Trajectory ID	-	Generated
Timestamp	-	ADS-B
Time	-	ADS-B
Altitude	ft	ADS-B
Latitude	deg	ADS-B
Longitude	deg	ADS-B
Ground speed	kts	ADS-B
Track	deg	ADS-B
Rate of Climb	ft/min	ADS-B
Temperature	Kelvin	ECMWF - ERA5
u, v wind velocities	m/s	ECMWF - ERA5
Vertical air velocity	Pa/s	ECMWF - ERA5

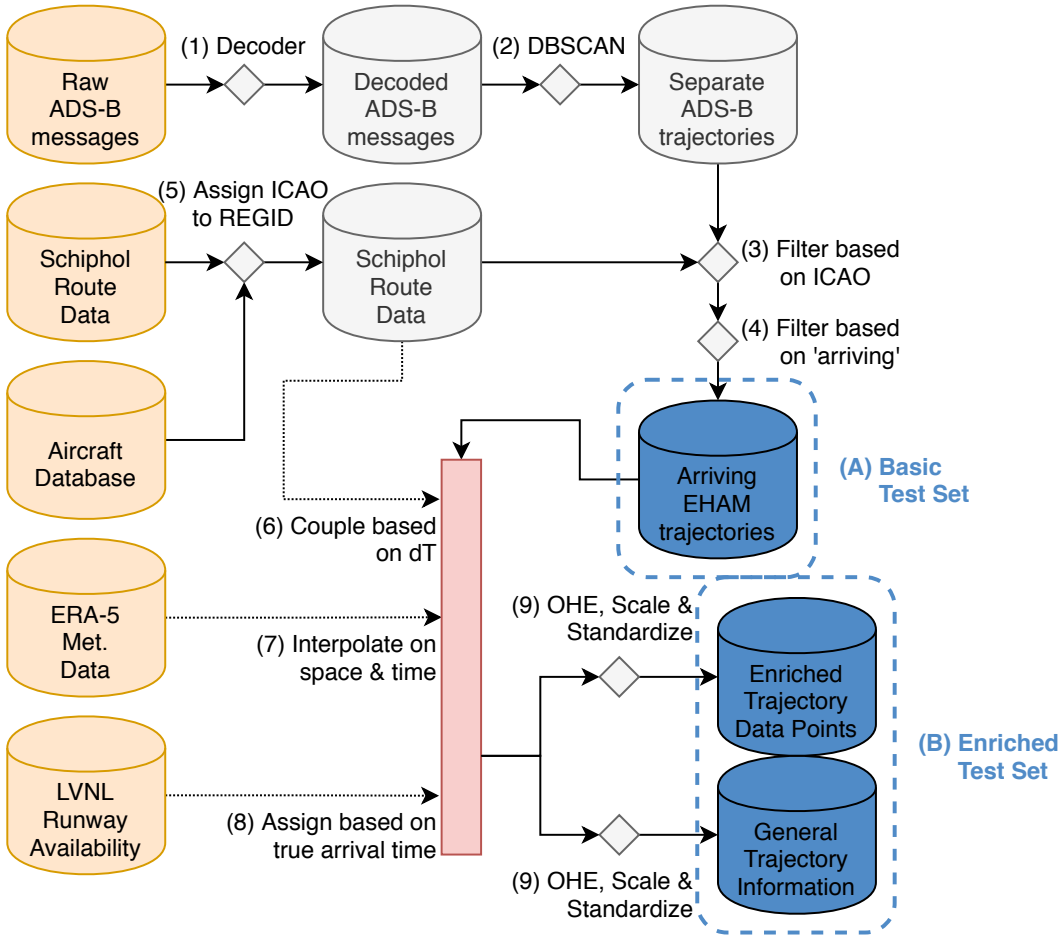


Figure 3.1: Data processing flow from data source to the final data sets. Orange depicts input data, diamonds depict actions on the data, blue represents final data sets and the red bar represents 3 available actions on the data which can be combined in any way desired.

Table 3.8: Data available on every single trajectory as a whole

Parameter	Unit	Source
Trajectory ID	-	Generated
A/C REGID	-	Schiphol
A/C Model	-	Schiphol
Operator	-	Schiphol
Flightno	-	Schiphol
Weekday	-	Schiphol
Timing of the flight	-	Schiphol
Planned Arrival time	-	Schiphol
Final Arrival time	-	Schiphol
Delay	-	Schiphol
Origin / Destination	-	Schiphol
Runway Availability at ETA	-	LVNL

Table 3.9: Number of trajectories available after each data processing step, over the period of 15-10-2019 to 20-01-2020

Step in data processing and coupling	Number of trajectories	Success Rate
Extracted trajectories for all ICAOs visting EHAM	217000	-
Routes extracted from EHAM website	54500	-
Arriving trajectories extracted from ADS-B Data	48645	89.3%
Coupled trajectory-route pairs	41760	85.8%
Overall coupling rate	**-**	**76.6%**

3.10 Data Analysis

Between 15-10-2019 and 20-01-2020, a period of approximately 3 months, all data sources presented before have been used in order to construct a data set with the structure as presented in section 3.9. It consists of about 41,700 arriving flights and in this section an overview of the trajectories available is presented. It should be noted that the numbers presented in the remainder of this section do not represent all arrivals at EHAM over the specified period, as the trajectories are already filtered and outliers are removed. The number of trajectories available after each processing step can be found in table 3.9. The trajectories omitted from the database are not considered as they are removed by either the DBSCAN algorithm forming the trajectories or by the limits set for an arriving aircraft in section 3.7.2. The roughly 25% of the total which is omitted are therefore considered to be outliers.

Now, the remaining trajectories are analysed. First of all, the aircraft types present in the data are shown in fig. 3.2. Here, it can be seen that several aircraft types dominate the airport. These aircraft types are also expected to yield the best results seeing as the main share of historical data, on which the predictors are trained, will also originate from these aircraft. From the first experiments it should be concluded whether or not the aircraft types that arrive at EHAM less frequently can be predicted as the performance of the predictor may be too low for these.

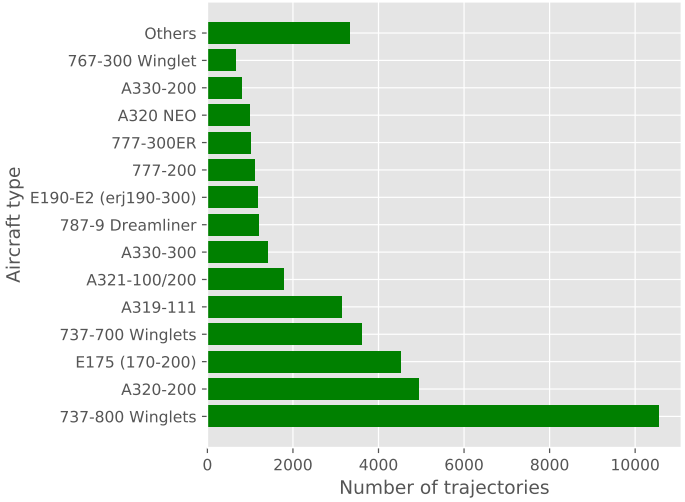


Figure 3.2: Total number of arriving trajectories available per a/c type from 15-10-2019 to 20-01-2020

Secondly, the operator or airline of each trajectory was counted, resulting in fig. 3.3. Here, it can be seen that the flag carrier of the Netherlands, KLM, has a large presence. The influence of the operator on the performance of a predictor is to be investigated, but it is expected that the predictions for KLM will yield the best results seeing as the largest share of historical data is available for this operator.

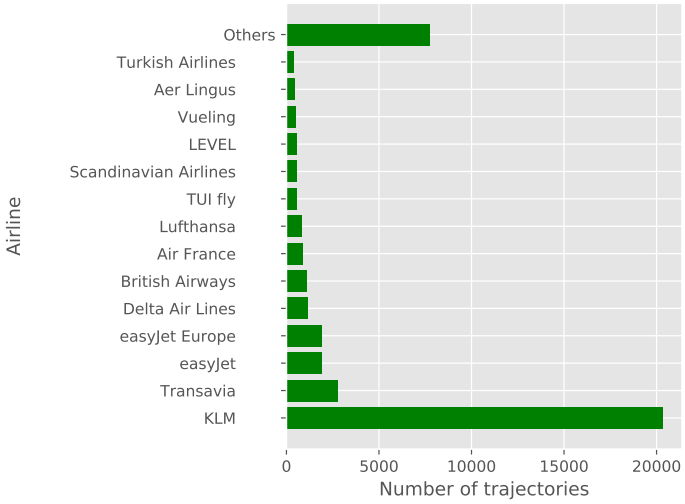


Figure 3.3: Total number of arriving trajectories available per operator from 15-10-2019 to 20-01-2020

The trajectories per day of the week are more evenly spread, as can be seen in fig. 3.4. Whether the parameter 'weekday' has influence on the performance of a predictor is yet to be tested during the experiments. However, it is not expected that the performance will be better for a trajectory on Monday versus one on Sunday as the amount of historical data available is comparable.

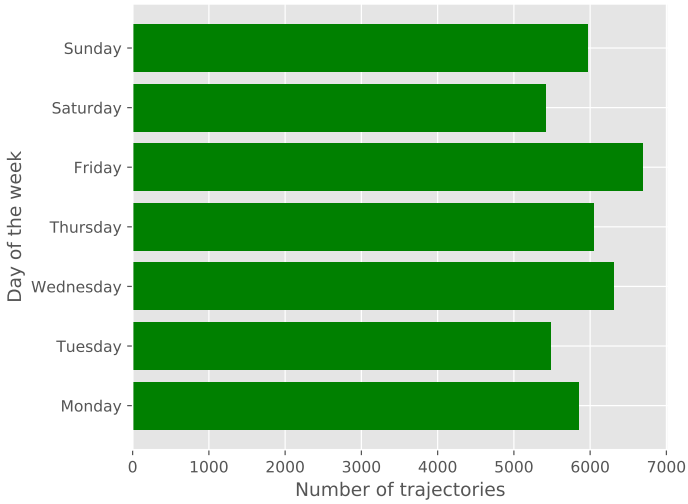


Figure 3.4: Total number of arriving trajectories available per day of the week from 15-10-2019 to 20-01-2020

Furthermore, the origins of the trajectories are determined. From this, it can be concluded that these are spread over a large number of airports, with the top 15 origins accounting for 600 to 1200 trajectories each and the other origins accounting for another 30,050 trajectories. It is expected that, if the origin is of influence to the predictor as a parameter, origins with a larger number of trajectories available will perform better in the predictors than origins with a significantly lower number of trajectories in the historical database.

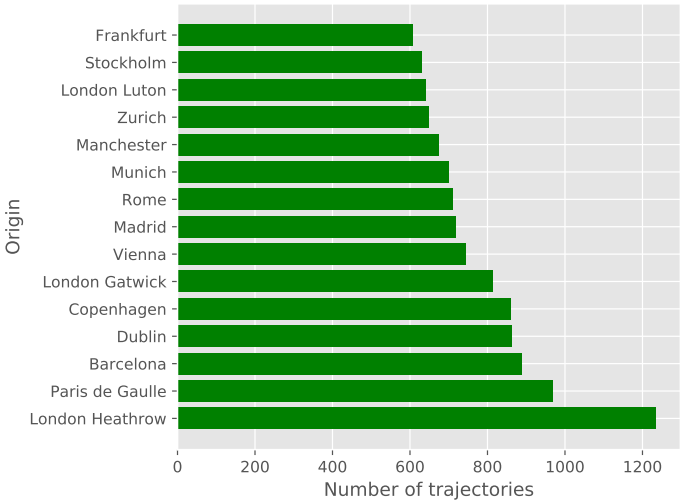


Figure 3.5: Number of arriving trajectories available per origin from 15-10-2019 to 20-01-2020 for the top 15 origins

4

Experiment Setup

In this chapter the next phase of this study will be outlined. During this phase the experiments will be carried out which will in turn answer the research questions presented in the introduction. First, an introduction will be given to the baseline experiment in section 4.1, after which the main experiment phase will be presented in section 4.2. The evaluation of these experiments is detailed in section 4.3 and the validation procedures in section 4.4. Finally, the resources required for the experiments are presented in section 4.5 and the limitations of this approach in section 4.6.

4.1 Baseline experiment

The baseline simulations are carried out using BlueSky. This simulation only has to be performed once as only one source of input data is required and only one method will be tested. In order to obtain the baseline, the experiment consists of three steps, which are defined as follows. This process is also shown in fig. 4.1

1. Provide aircraft specifications and waypoints to BlueSky
2. Perform simulation from entry point to EHAM
3. Log trajectory from the simulation

The first step entails initialising the simulation by providing the details required for the simulation. The second step is the simulation of the flight and the last step is logging the executed 4D trajectory. The simulations for each trajectory can be performed simultaneously and do not depend on for example the time of the day as no interaction between aircraft is taken into account. According to the processing power required and available the trajectories will be split in a number of batches for this baseline experiment.

4.2 Main experiment

The final goal of the main experiments is to provide trajectory predictions from each selected predictor and being able to evaluate them on the stated terms. Multiple experiments will be carried out, each on different data sets consisting of a specific combination of data sources and using a unique split of

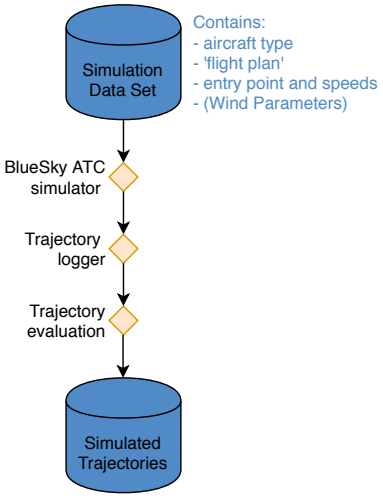


Figure 4.1: Schematic overview of the baseline experiment

training and testing data. However, regardless of the input data, each experiment will consist of 6 phases which are defined as follows. A schematic overview can be found in fig. 4.2.

1. Training the PCA model
2. Training the DBSCAN model
3. Training a predictor (LWLR,GBM,MLP,LSTM) on each cluster
4. Perform PCA on test data using the trained PCA model
5. Assigning the test data to a cluster using the trained DBSCAN model
6. Predict new trajectories using the trained predictor

During steps 1 and 2, first the PCA and DBSCAN algorithms will be trained for the specific data set at hand using the training set, after which the trained algorithms are applied on the testing set in steps 4 and 5. This is necessary as the PCA algorithm needs to learn which 'categories' are available and thus where to map new data on. Similarly, the DBSCAN algorithm needs to define which clusters are available based on the training set, after which it can assign a new trajectory to an existing cluster from the testing set. Steps 1 and 2 have to be performed only once once during each experiment, while steps 4 and 5 are performed at each timestep in the prediction. It is of importance, especially during the first few timesteps of a trajectory, to re-evaluate the cluster a trajectory belongs to during the flight. The clustering will be based on the eucledian distances between the 3D spatial trajectories, not taking other parameters into account. DBSCAN does not offer a standard way to assign new trajectories to a cluster, especially seeing as in our case in the beginning of a trajectory only a short part of this trajectory is available. It is chosen to assign the trajectories using a k-Nearest Neighbours search, identifying the training trajectories closest to the to be tested trajectory and assigning it to the closest training trajectory. Seeing as this can differ over time, as trajectories move away or towards eachother, this should be repeated every timestep.

Then, each predictor is trained in step 3 on the training data once per experiment and is used to perform the trajectory prediction at each timestep. An overview of how these predictions are evaluated will be given in the next section.

This process details the procedures of each single experiment. However, multiple experiments are to be executed in order to assess the performance of each predictor, each set of input data parameters and also to validate the results obtained using these combinations. Therefore, each predictor is validated

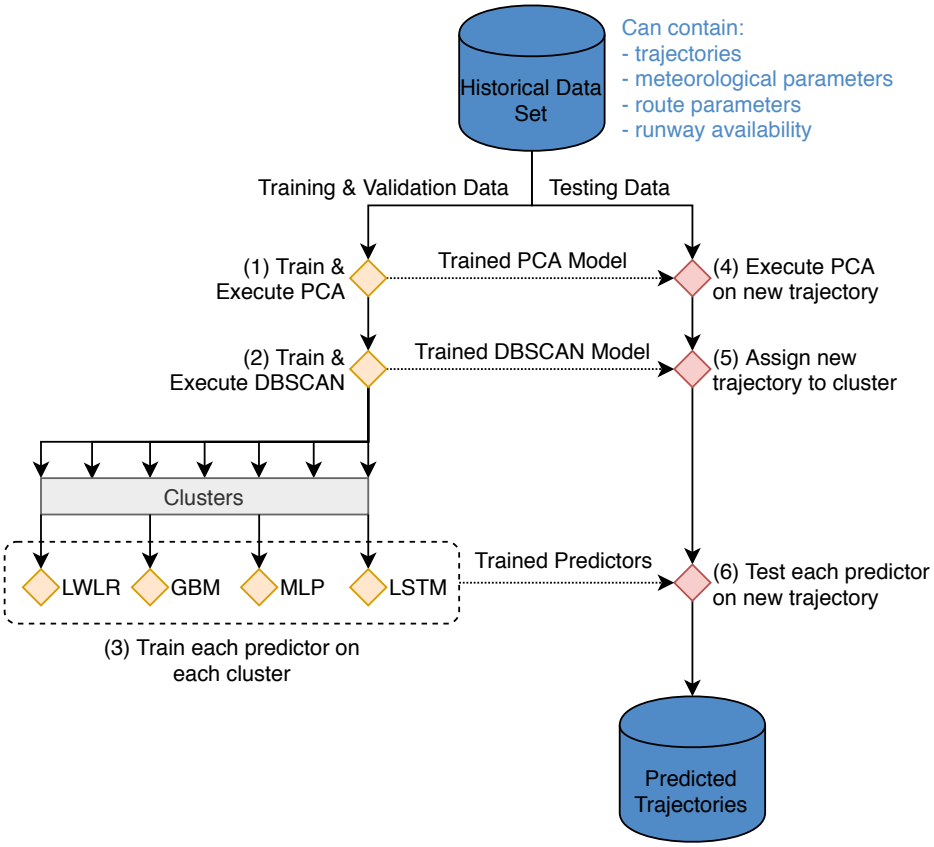


Figure 4.2: Schematic overview of one single experiment

as described in section 4.4, resulting in a set of experiments. The influence of the input parameters on the predictions are to be investigated before performing the main experiment. Therefore, using one predictor, this is tested by performing the experiments using specific sets of input parameters. These combinations can be found in table 4.1 and the rationale is as follows. First, the performance of the ADS-B position parameters only is tested, after which each parameter is added once in order to test the predictive performance individually. Then, combinations of parameters are made which are assumed to have a correlation between them. For example, the wind and runway availability are expected to have a similar influence on the routing of the aircraft, as the wind is the main indicator of which runways are open and closed. After performing these tests, it is decided whether or not a parameter is redundant and then the remainder of the experiments are conducted using one set of input parameters, which at most can contain all the previously mentioned parameters. It would be optimal to perform tests using each combination of input parameters, however if the parameters mentioned in conditions 2 to 9 are to be combined in each possible way, $2^8 = 256$ experiment conditions would arise. Testing all these is not feasible in the time available and therefore the usefulness of each parameter is only tested using the experiments stated in the table. Also, previous research has focused entirely on the influence of the input data on the predictions and this is not the main goal of this research. After this analysis each predictor is tested using the same set of input parameters in order to be able to compare them properly.

4.3 Experiment evaluation

In order to answer the research questions posed in chapter 1, several metrics are taken into account. Therefore, each experiment is evaluated to base future conclusions on. In section 2.4 it is described which quality assessments are to be used and why. First of all, the speed of the predictor is measured in two ways. During each experiment, the time to train each model is noted. This includes the PCA,

Table 4.1: Input parameters for each experiment evaluating the influence of the input parameters on the prediction performance

Experiment condition #	Input parameters
1	ADS-B position only
2	ADS-B position and ADS-B speeds
3	ADS-B position and Temperature & Winds
4	ADS-B position and Runway availability
5	ADS-B position and Origin
6	ADS-B position and a/c type
7	ADS-B position and Airline
8	ADS-B position and Day of week
9	ADS-B position and Time of the day (planned arrival time)
10	ADS-B position and Temperature, Winds and Runway availability
11	ADS-B position and a/c type & Airline
12	ADS-B position and a/c type & Origin
13	ADS-B position and ADS-B speeds and a/c type
14	ADS-B position and Origin and Day of the week
15	ADS-B position and Origin and Time of the day

Table 4.2: Evaluation metrics of the main experiment

Metric	Unit	Evaluation moment
Absolute horizontal error	m	Every timestep
Absolute altitude error	m	Every timestep
ETA error	sec	Every timestep
Stability of the predictor	sec	Every timestep
Computation time training phase	sec	Training phase
Computation time predicting phase	sec	Every timestep

DBSCAN and predictor models. Then, the time is noted to perform the predictions using each individual predictor. By standardising these values these will be independent from the hardware used and relative to each other.

The stability of the predictor is tested by comparing the prediction at each time step with the previous prediction in terms of ETA. This results in a delta time which can later be used to assess the stability. Also, the stability is tested by considering the time delta between 10 predictions as if the time delta is small each time, but in the same direction, this can lead to a large offset over a longer period of time. The accuracy of the predictors is assessed by comparing the 4D trajectory at each time step both with the baseline prediction as with the true trajectories from the database. In order to assess at which look-ahead time the predictors provide meaningful predictions, this is also assessed at each time step. The accuracy is also evaluated in a temporal manner, meaning that the predicted arrival time will be compared with the true arrival time. An overview of these metrics can be found in table 4.2

4.4 Validation

In order to ensure the predictors are tested in a correct way and to compare them amongst each other, the results of the prediction should be validated. A straightforward method often applied in the field of machine learning is a so-called hold-out procedure in which a part of the data set is not used for

training the data in order for the model to be tested on unseen data afterwards. A common split is 70% training data and 30% testing data. However, it cannot be said with certainty that the training set is a representative sample of the full data set if split this way. Also, it will only provide a point estimate of the accuracy.

To overcome these disadvantages, cross validation can be used. This entails partitioning the data into K parts and then performing K tests in which each part is the testing test once and part of the training set K-1 times. Performing validation this way provides a more representative outcome and it provides an accuracy distribution instead of a point estimate [9]. This allows for providing a confidence interval of the accuracy. Finally, seeing as we will be comparing multiple models and multiple sets of parameters, these should also be tuned and the choices validated [9]. For this purpose, nested cross validation will be used. This entails the process described before, with another cross validation one layer deeper at each step. This process is shown in fig. 4.3. Finally, seeing as the predicted trajectories are validated against the true flown trajectories extracted from the ADS-B data, they can be considered fully validated after this procedure.

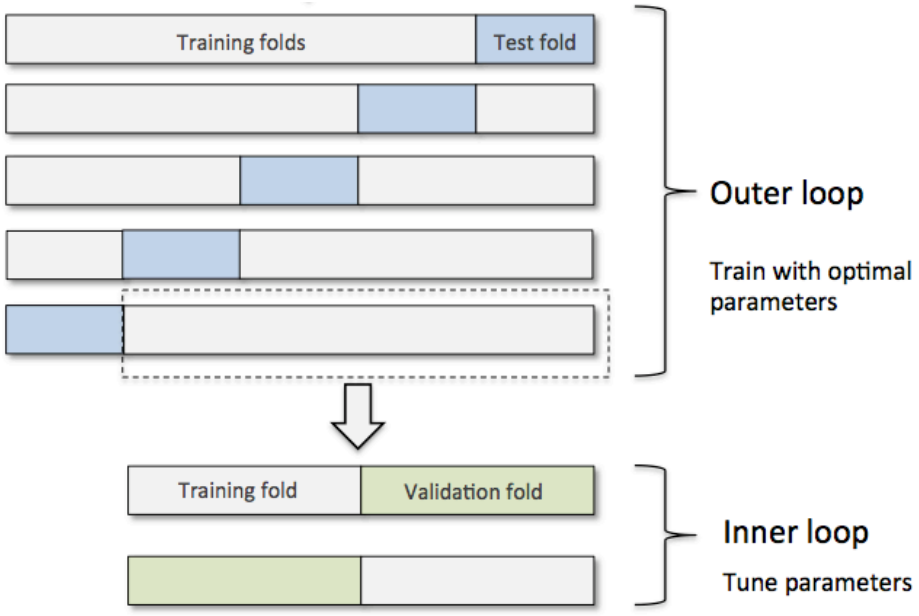


Figure 4.3: Schematic of nested cross validation, showing the cross validation on the first level, with unique testing sets for each run and the nested deeper layer[1]

4.5 Resources

The experiments described above will have to be conducted using software. For the baseline experiments, python-based BlueSky is used, which is openly accessible and developed at the TU Delft. For the main experiments to be conducted also python (3.x) will be used, extended with amongst others the pandas package for data handling and the scikit-learn package for machine learning purposes [43].

BlueSky is a tool for air traffic simulations [27], which is able to simulate the behaviour of aircraft and can execute flight plans using commercial (BADA) or open source (OpenAP) aircraft performance parameters as an input. Simulating the flights in such a simulator is a close approximation of current trajectory prediction methods, as the latter perform the same trajectory calculations using aircraft performance models as the simulator. Therefore, it is chosen to use the simulator results to serve as a baseline representing the current methods used by ATC. For each batch of aircraft, a traffic file is loaded which contains the required information, which are the aircraft type, location, altitude, velocities, the

[1] https://sebastianraschka.com/faq/docs/evaluate-a-model.html

waypoints and destination. Using these inputs, BlueSky will simulate the trajectory of the aircraft using the APMs, which will then be logged, resulting in a series of points in time and 3D space, representing the flown trajectory. These simulations will serve as a baseline to compare the other predictions with.

Due to the amount of combinations of data available, the four different machine learning predictors to be tested and the nested cross validation procedure, testing the machine learning predictors will be the most time consuming part of the study. This should be kept in mind and as soon as the first experiments are conducted it should be re-evaluated whether this amount of experiments is feasible in the remaining time available. However, due to the nature of the experiments, which are computer-based, this is not expected to be a problem for this research. The simulations can run simultaneously and no major adaptions to the models have to be performed to test the different combinations of input data.

4.6 Limitations

The main limitation of this experiment approach is that no real-life tests involving air traffic controllers are used while their interaction with a predictor may influence the performance of a predictor. For example, if ATC would adapt their planning behaviour based on the information provided by the enhanced predictor, the predictions in turn would be less accurate. However, as this is a first step in the developments of these tools, testing with ATC interaction should be considered after these tests have proven to be a success. Similarly, no interaction between predicted trajectories is present in these experiments, whereas if two 4D trajectories cross, it can be expected that these trajectories will change due to ATC intervention. This is, however, considered out of the scope of this research.

Another limitation is that no true flight plan data is available and that therefore the baseline predictions using BADA may not be as accurate as available to ATC in existing systems. Therefore, the simulations using BlueSky may not be an accurate representation of current ATC systems. This approach does provide opportunities to test additional machine learning based predictors and additional types of input data. However the time available to perform this study is limited and it is therefore chosen to test only the aforementioned methods and inputs, limiting the scope of the research. The planning of the next phase of this study is presented in the next chapter.

5

Planning

In order to provide answers to the research questions posed in chapter 1 the experiments have to be executed in time. Therefore, a thorough planning including all steps required to obtain the results is made. First, the main steps to be taken are identified in the first section, after which a brief planning is proposed in section 5.2.

5.1 Steps to be taken

Currently, the data processing has been developed and every day new data is added to the database. Although the input data is not complete yet, experiments can already be performed using the data available. Therefore, the next steps to be taken are the development of all models used in the experiments and introduced in section 2.2. The order in which this has to be done is of importance.

First of all, the PCA and DBSCAN models have to be implemented using the python packages available as these are a prerequisite to each experiment. Then, it is chosen to first develop one predictor such that a first experiment can be conducted. This way, the methodology can be tested and it can be re-evaluated whether this is the correct approach for the remainder of the study. Then, while the first predictor is being tested, the other predictors can be coded and developed, after which these can also be used in experiments. It is chosen to first implement MLP as a predictor seeing as it is a proven algorithm for trajectory prediction (section 2.2). Also, the first experiment will be executed with a relatively easy data set. An overview of the steps to be taken can be found in the following list:

1. Implement PCA & DBSCAN algorithms
2. Implement first predictor: MLP
3. Conduct 'pilot' experiment using a dataset enriched with meteorological and O/D data only
4. Evaluate and adapt methodology
5. Implement all predictors
6. Conduct all experiments

At the same time, the baseline method can be developed. Although the pilot experiment can be validated using the true flown trajectories, a comparison with the baseline method can help in evaluating the methodology. The steps to be taken to obtain the baseline are as follows:

1. Obtain input for BlueSky simulations
2. Conduct a test baseline experiment
3. Evaluate and adapt methodology
4. Conduct full baseline experiment

5.2 Timeline

The steps presented in the previous section have to be planned well in order to obtain the results by the end of this study. For this purpose, several milestones have been planned, which are defined as follows. After these milestones all results are collected which need to be documented and presented.

1. **PCA & DBSCAN implemented:** 01/03/2020
2. **Input ready for baseline simulations:** 01/03/2020
3. **MLP implemented:** 09/03/2020
4. **'Pilot' experiments, both baseline and MLP conducted:** 12/03/2020
5. **All predictors implemented:** 30/03/2020
6. **All experiments conducted:** 20/04/2020

6

Conclusions

This report details the literature study into the field of 4D trajectory prediction for aircraft using various techniques and the prerequisites for such predictors. It is concluded that in order to advance the field of 4D trajectory prediction, various machine learning methods should be tested. These tests are to be evaluated based on the accuracy, speed and stability compared to existing systems and the true flown trajectories. The baseline method representing the currently existing systems is chosen to be a simulation in BlueSky using the standard aircraft performance models and parameters and generated flight plan data as no source is available for the latter.

For the new predictors it is chosen to use a combination of clustering the aircraft trajectories and then predict them using another algorithm. For clustering it is concluded to use the density based DBSCAN method as it performs well for data containing outliers and has a proven track record in recent literature. In order to speed up the clustering process, the data is first processed using Principal Component Analysis, which reduces the dimensions of the data.

The predictors will be based on four different machine learning methods. It is concluded that Locally Weighted Linear Regression has proven to be the best performing regression algorithm, while Gradient Boosting Machines is the most promising Decision Trees algorithm. In the field of neural networks it is concluded to compare deep feed forward neural networks with recurrent neural networks as this has not been done before. For the latter, Long Short-Term Memory neural networks are found to be the most suitable option from literature.

The main input data required for this study was found to be ADS-B data to represent the trajectories. These trajectories are then enriched using meteorological data, route data and runway availability in order to enhance the performance of the predictors. In the experiments, the influence of each parameter is to be tested. For the current sourcing period from 15-10-2019 to 20-01-2020 about 41,700 trajectories were found and processed, after outlier detection. These were coupled to the route data based on the final location and time of each trajectory, resulting in a set of data large enough and coupled to all data sources and which is ready to perform experiments with.

The experiments are to be conducted in the next phase of this study and it is proposed to first implement and fully execute the experiments for one predictor in order to assess the performance and speed of the experiment set-up. Based on this assessment, it can be chosen to execute the full experiment plan or to reduce the number of experiments if time will become a constraint.

To conclude, the preparations for the experiments in this study have been conducted, including a literature study, data sourcing, data preparation and processing and data coupling. A plan has been developed which can be used to conduct the experiments and the evaluation and validation strategy is defined. The next step is to conduct the experiments in order to answer the research questions.

Bibliography

[1] R. Alligier, D. Gianazza, and N. Durand. Learning the aircraft mass and thrust to improve the ground-based trajectory prediction of climbing flights. *Transportation Research Part C: Emerging Technologies*, 36:45–60, 2013. ISSN 0968090X. doi: 10.1016/j.trc.2013.08.006. URL `http://dx.doi.org/10.1016/j.trc.2013.08.006`.

[2] Richard Alligier. *Machine Learning Applied to Aircraft Trajectory Prediction*. PhD thesis, l'Institut National Polytechnique de Toulouse (INP Toulouse), 2014.

[3] Richard Alligier, David Gianazza, and Nicolas Durand. Machine Learning Applied to Airspeed Prediction During Climb. In *ATM Seminar. 11th USA/EUROPE Air Traffic Management R&D Seminar, FAA & Eurocontrol, Jun 2015, Lisboa, Portugal.*, 2015.

[4] Richard Alligier, David Gianazza, and Nicolas Durand. Machine Learning and Mass Estimation Methods for Ground-Based Aircraft Climb Prediction. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3138–3149, 2015. ISSN 15249050. doi: 10.1109/TITS.2015.2437452. URL `https://hal-enac.archives-ouvertes.fr/hal-01181173`.

[5] Richard Alligier, David Gianazza, and Nicolas Durand. Predicting Aircraft Descent Length with Machine Learning. In *7th International Conference on Research in Air Transportation (ICRAT)*, 2016. URL `https://hal-enac.archives-ouvertes.fr/hal-01353960`.

[6] Samet Ayhan and Hanan Samet. Aircraft Trajectory Prediction Made Easy with Predictive Analytics. In *22nd ACM SIGKDD International Conference*, pages 21–30, 2016. ISBN 9781450342322. doi: 10.1145/2939672.2939694.

[7] Samet Ayhan and Hanan Samet. Time Series Clustering of Weather Observations in Predicting Climb Phase of Aircraft Trajectories. In *ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pages 25–30, Maryland, USA, 2016. ISBN 9781450345774. doi: 10.1145/3003965.3003968. URL `http://dx.doi.org/10.1145/3003965.3003968`.

[8] Shane T. Barratt, Mykel J. Kochenderfer, and Stephen P. Boyd. Learning Probabilistic Trajectory Models of Aircraft in Terminal Airspace From Position Data. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–10, 2018. ISSN 1524-9050. doi: 10.1109/TITS.2018.2877572. URL `https://ieeexplore.ieee.org/document/8551278/`.

[9] Avrim Blum, Adam Kalai, and John Langford. Beating the hold-out: bounds for K-fold and progressive cross-validation. *Proceedings of the Annual ACM Conference on Computational Learning Theory*, pages 203–208, 1999.

[10] Facundo Bre, Juan M. Gimenez, and Víctor D. Fachinotti. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings*, 158(November):1429–1441, 2018. ISSN 03787788. doi: 10.1016/j.enbuild.2017.11.045.

[11] Leo Breiman. Bagging Predictors. *Machine Learning*, 24(2):123–140, 1996. ISSN 08856125.

[12] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification And Regression Trees*. CHAPMAN & HALL/CRC, 1984. ISBN 9780412048418.

[13] Taoya Cheng, Deguang Cui, and Peng Cheng. Data mining for air traffic flow forecasting: A hybrid model of neural network and statistical analysis. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 1:211–215, 2003. doi: 10.1109/ITSC.2003.1251950.

[14] Mayara Condé, Rocha Murça, R John Hansman, Hamsa Balakrishnan, Richard Delaura, Richard Jordan, and Tom Reynolds. Trajectory Clustering and Classification for Characterization of Air Traffic Flows. In *16th AIAA Aviation Technology, Integration, and Operations Conference*, Reston, Virginia, 6 2016. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-440-4. doi: 10.2514/6.2016-3760. URL http://arc.aiaa.org/doi/10.2514/6.2016-3760.

[15] Arjen de Leege, Marinus van Paassen, and Max Mulder. A Machine Learning Approach to Trajectory Prediction. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 1–14, Delft, 2013. doi: 10.2514/6.2013-4782.

[16] Marco Enriquez. Identifying Temporally Persistent Flows in the Terminal Airspace via Spectral Clustering. Technical report, 2013. URL http://atmseminar.org/seminarContent/seminar10/papers/183-Enriquez_0124131201-Final-Paper-4-12-13.pdf.

[17] Martin Ester, Hans-Peter Kriegel, Jiirg Sander, and Xiaowei Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, volume 96, 1996. URL www.aaai.org.

[18] Esther Fernández Calvo, José Manuel Cordero, George Vouros, Nikos Pelekis, Theocharis Kravaris, Georgiou Harris, Georg Fuchs, Natalia Andrienko, Gennady Andrienko, Enrique Casado, David Scarlatti, and Pablo Costas. DART: A Machine-Learning Approach to Trajectory Prediction and Demand-Capacity Balancing. *Seventh SESAR Innovation Days*, (November), 2017.

[19] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5):1189–1232, 2001. ISSN 00905364. doi: 10.2307/2699986.

[20] Jerome H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38(4):367–378, 2002. ISSN 01679473. doi: 10.1016/S0167-9473(01)00065-2.

[21] Maxime Gariel, Ashok N. Srivastava, and Eric Feron. Trajectory clustering and an application to airspace monitoring. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1511–1524, 2011. ISSN 15249050. doi: 10.1109/TITS.2011.2160628.

[22] Areski Hadjaz, Gaétan Marceau, Pierre Savéant, and Marc Schoenauer. Online Learning for Ground Trajectory Prediction. In *SESAR 2nd Innovation Days*, 2012. URL http://arxiv.org/abs/1212.3998https://pdfs.semanticscholar.org/d0cd/1dc961f10d912349b3fdfbc23d6a6ca6b08d.pdf.

[23] Mohammad Ghasemi Hamed, David Gianazza, Mathieu Serrurier, and Nicolas Durand. Statistical prediction of aircraft trajectory: regression methods vs point-mass model. *10th USA/Europe ATM R&D Seminar*, 2013(June):pp, 2013. URL https://hal-enac.archives-ouvertes.fr/hal-00911709.

[24] Sarah L. Harris and David Money Harris. 3 - Sequential Logic Design. In *Digital Design and Computer Architecture*, pages 108–171. Morgan Kaufmann, arm editio edition, 2016. ISBN 9780128000564. URL https://doi.org/10.1016/B978-0-12-800056-4.00003-0.

[25] Andrés Muñoz Hernández, Enrique J.Casado Magaña, Antonio Gracia Berna, Andres Munoz Hernandez, Enrique J. Casado Magana, and Antonio Gracia Berna. Data-driven aircraft trajectory predictions using ensemble meta-estimators. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 2018-Septe:1–10, 9 2018. ISSN 21557209. doi: 10.1109/DASC.2018.8569535. URL https://ieeexplore.ieee.org/document/8569535/.

[26] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780, 1997. ISSN 08997667. doi: 10.1162/neco.1997.9.8.1735.

[27] Jacco M. Hoekstra and Joost Ellerbroek. BlueSky ATC simulator project: an open-data and open-source approach. *Proceedings of the 7th International Conference on Research in Air Transportation*, pages 1–8, 2016.

[28] Sungkwon Hong and Keumjin Lee. Trajectory Prediction for Vectored Area Navigation Arrivals. *Journal of Aerospace Information Systems*, 12(7), 2015. doi: 10.2514/1.I010245.

[29] Marko Hrastovec. Prediction of aircraft trajectories for air traffic control using machine learning approaches. Technical report, 2018. URL `http://eprints.fri.uni-lj.si/4135/1/24011162-MARKO_HRASTOVEC-Napovedovanje_letalskih_trajektorij_za_potrebe_kontrole_zračnega_prometa_s_pristopi_strojnega_učenja.pdf`.

[30] Marko Hrastovec and Franc Solina. Machine learning model for aircraft performances. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, pages 41–8, 2014. ISSN 21557209. doi: 10.1109/DASC.2014.6979541.

[31] Marko Hrastovec and Franc Solina. Prediction of aircraft performances based on data collected by air traffic control centers. *Transportation Research Part C: Emerging Technologies*, 73(December): 167–182, 2016. ISSN 0968090X. doi: 10.1016/j.trc.2016.10.018. URL `http://dx.doi.org/10.1016/j.trc.2016.10.018`.

[32] Christopher Jesse, Honghai Liu, Edward Smart, and David Brown. Analysing Flight Data Using Clustering Methods. Technical report, 2008. URL `https://link.springer.com/content/pdf/10.1007/978-3-540-85563-7_92.pdf`.

[33] Yann Le Fablec and Jean Marc Alliot. Using Neural Networks to predict aircraft trajectories. Technical report, 1999. URL `http://alliot.info/papers/icai99.pdfhttp://www.amisducena.fr/public/NL99-008.pdf`.

[34] Kenneth Leiden and Stephen Atkins. Trajectory Clustering for Metroplex Operations. In *11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, number September, pages 1–15, 2011. doi: 10.2514/6.2011-7066.

[35] Yi Lin, Jian-Wei Zhang, and Hong Liu. An algorithm for trajectory prediction of flight plan based on relative motion between positions. *Frontiers of Information Technology & Electronic Engineering*, 19(7):905–916, 2017. ISSN 2095-9184. doi: 10.1631/fitee.1700224. URL `https://doi.org/10.1631/FITEE.1700224`.

[36] Yulin Liu and Mark Hansen. Predicting Aircraft Trajectories: A Deep Generative Convolutional Recurrent Neural Networks Approach. Technical report, 2018. URL `https://arxiv.org/pdf/1812.11670v1.pdf`.

[37] Ioannis Lymperopoulos, John Lygeros, and Andrea Lecchini. Model Based Aircraft Trajectory Prediction during Takeoff. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Reston, Virigina, 8 2006. American Institute of Aeronautics and Astronautics. ISBN 978-1-62410-046-8. doi: 10.2514/6.2006-6098. URL `http://arc.aiaa.orghttp://arc.aiaa.org/doi/10.2514/6.2006-6098`.

[38] Stéphane Mondoloni and Ibrahim Bayraktutar. Impact of factors, conditions and metrics on trajectory prediction accuracy. In *Proceedings of the 6th USA/Europe Air Traffic Management Research and Development Seminar, ATM 2005*, pages 305–314, 2005. ISBN 9781510801707.

[39] Stéphane Mondoloni, Sipke Swierstra, and Mike Paglione. Assessing trajectory prediction performance - Metrics definition. *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, 1:1–13, 2005. doi: 10.1109/DASC.2005.1563347.

[40] Florence Nicol. Functional principal component analysis of aircraft trajectories. In *2nd International Conference on Interdisciplinary Science for Innovative Air Traffic Management*, 2013. URL `https://hal-enac.archives-ouvertes.fr/hal-00867957`.

[41] Mike M Paglione and Robert D Oaks. Implementation and Metrics for a Trajectory Prediction Validation Methodology. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007. doi: 10.2514/6.2007-6517. URL `http://arc.aiaa.org`.

[42] Yutian Pang, Nan Xu, and Yongming Liu. Aircraft Trajectory Prediction using LSTM Neural Network with Embedded Convolutional Layer. In *Proceedings of the Annual Conference of the PHM Society*, 2019. doi: 10.36001/phmconf.2019.v11i1.849. URL https://doi.org/10.36001/phmconf.2019.v11i1.849.

[43] Fabian Pedregosa, Ron Weiss, and Matthieu Brucher. Scikit-learn : Machine Learning in Python. 12:2825–2830, 2011.

[44] Frank Rehm. Clustering of Flight Tracks. *American Institute of Aeronautics and Astronautics*, 2010. doi: 10.2514/6.2010-3412. URL http://arc.aiaa.org.

[45] Julia Rudnyk, Joost Ellerbroek, and Jacco Hoekstra. Trajectory Prediction Sensitivity Analysis Using Monte Carlo Simulations. In *Aviation Technology, Integration, and Operations Conference*, 2018. doi: 10.2514/6.2018-3669.

[46] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[47] Hollis F Ryan, Mike M Paglione, Faa J William, and Steven M Green. Review Of Trajectory Accuracy Methodology And Comparison Of Error Measurement Metrics. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004. doi: 10.2514/6.2004-4787. URL http://arc.aiaa.org.

[48] Erwan Salaün, Maxime Gariel, Adan E. Vela, and Eric Feron. Aircraft proximity maps based on data-driven flow modeling. *Journal of Guidance, Control, and Dynamics*, 35(2):563–577, 2012. ISSN 07315090. doi: 10.2514/1.53859.

[49] F.; Desart B. Sillard, L.; Vergne. EUROCONTROL Synonym Aircraft Report for the Base of Aircraft Data (BADA) - Revision 3.7. Technical report, 2009. URL http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:REvisiting+the+Swiss+Cheese+model+of+accidents#0.

[50] L. Sillard, F. Vergne, and B. Desart. *User Manual For The Base Of Aircraft Data (Bada) Revision 3.12*, volume 12. 2014. ISBN 0014-0139. URL http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:REvisiting+the+Swiss+Cheese+model+of+accidents#0.

[51] Steve Green Stephane Mondoloni, Mike Paglione. Trajectory Modeling Accuracy For Air Traffic Management Decision Support Tools. In *ICAS Congress*, 2002.

[52] Junzi Sun. SIL: Python Client for Mode-s Beast Raw Data, 2019. URL https://github.com/junzis/sil.

[53] Junzi Sun. *The 1090MHz Riddle*. TU Delft, accessed o edition, 2019. ISBN 0000100110. URL https://mode-s.org/decode/adsb/introduction.html.

[54] Junzi Sun, Joost Ellerbroek, and Jacco Hoekstra. Flight Extraction and Phase Identification for Large Automatic Dependent Surveillance–Broadcast Datasets. *Journal of Aerospace Information Systems*, 14(10):566–571, 2017. ISSN 1940-3151. doi: 10.2514/1.i010520.

[55] Junzi Sun, Joost Ellerbroek, and Jacco M Hoekstra. OpenAP : The Open-Source Aircraft Performance Model and Associated Toolkit. Technical Report April, 2019.

[56] Noboru Takeichi, Ryosuke Kaida, Akihide Shimomura, and Takahiro Yamauchi. Prediction of Delay due to Air Traffic Control by Machine Learning. In *AIAA Modeling and Simulation Technologies Conference*, number January, 2017. doi: 10.2514/6.2017-1323.

[57] K Tastambekov, S Puechmorel, D Delahaye, and C Rabut. Aircraft trajectory forecasting using local functional regression in Sobolev space. *Transportation Research Part C*, 39, 2014. doi: 10.1016/j.trc.2013.11.013. URL http://dx.doi.org/10.1016/j.trc.2013.11.013.

[58] Christian Eduardo Verdonk Gallego, Víctor Fernando Gómez Comendador, Francisco Javier Sáez Nieto, Guillermo Orenga Imaz, and Rosa María Arnaldo Valdés. Analysis of air traffic control operational impact on aircraft vertical profiles supported by machine learning. *Transportation Research Part C: Emerging Technologies*, 95(March 2018):883–903, 2018. ISSN 0968090X. doi: 10.1016/j.trc.2018.03.017. URL `https://doi.org/10.1016/j.trc.2018.03.017`.

[59] Zhengyi Wang, Man Liang, and Daniel Delahaye. Short-term 4D Trajectory Prediction Using Machine Learning Methods. In *SID 2017, 7th SESAR Innovation Days, Nov 2017, Belgrade, Serbia*, 2017.

[60] Zhengyi Wang, Man Liang, and Daniel Delahaye. A hybrid machine learning model for short-term estimated time of arrival prediction in terminal manoeuvring area. *Transportation Research Part C: Emerging Technologies*, 95(January):280–294, 2018. ISSN 0968090X. doi: 10.1016/j.trc.2018.07.019. URL `https://doi.org/10.1016/j.trc.2018.07.019`.

[61] Zhengyi Wang, Man Liang, and Daniel Delahaye. Automated Data-Driven Prediction on Aircraft Estimated Time of Arrival. *Sesar Innovations Days 2018*, (December), 2018.

[62] Paul Weitz. Determination and visualization of uncertainties in 4D-trajectory prediction. *Integrated Communications, Navigation and Surveillance Conference, ICNS*, pages 1–9, 2013. ISSN 21554943. doi: 10.1109/ICNSurv.2013.6548525.

[63] Junfeng Zhang, Jie Liu, Rong Hu, and Haibo Zhu. Online four dimensional trajectory prediction method based on aircraft intent updating. *Aerospace Science and Technology*, 77:774–787, 2018. ISSN 12709638. doi: 10.1016/j.ast.2018.03.037. URL `https://doi.org/10.1016/j.ast.2018.03.037`.