DLR-IB-MO-HF-2021-62

A Novel Approach to Flight Phase Identification using Machine Learning

Masterthesis

Emy Arts







A Novel Approach to Flight Phase Identification using Machine Learning

Masterthesis at Research Group Knowledge Technology, WTM Prof. Dr. Stefan Wermter

> Department of Informatics MIN-Faculty Universität Hamburg

In cooperation with: **DLR** - German Aerospace Center Institute of Maintenance, Repair and Overhaul Prof. Dr. Hans Peter Monner Hein-Saß-Weg 22 21129 Hamburg Department of Process Optimisation and Digitalisation Dr. Ing. Florian Raddatz

submitted by: Emy Arts Course of study: Intelligent Adaptive Systems Matrikelnr.: 7132887 on 16.03.2021

Examiners: Prof. Dr. Stefan Wermter Dr. Annika Peters Adviser: Hendrik Meyer

Abstract

Flight phases are essential for many applications of aviation research. In this project, a novel machine learning model for the identification of flight phases is presented. The identification is performed on aircraft trajectory data, which in contrast to other flight data, is a publicly available resource obtained through the Automatic Dependent Surveillance Broadcast (ADS-B) concept. With the help of supervised simulation data, a model that aims at improving state-of-the-art flight phase identification on trajectory data is developed. The model combines Kmeans clustering, allowing the segmentation to capture transitions between phases more closely, with a Long Short-Term Memory (LSTM) network, able to learn the dynamics of a flight. The improvement of this work, compared with the state-ofthe-art model by Sun et al. [41] based on fuzzy logic, comprises: increasing the accuracy by more than 2%, adhering to the International Civil Aviation Organisation (ICAO) standard, and increasing the number of flight phases to include take-off, landing, and others. The latter shows potential, considering the performance on simulation data, however, requires more realistic training data to achieve similar performance on actual flights.

Contents

	Abs	ract IV	Τ
	\mathbf{List}	of Figures VI	Ι
	List	of Tables IX	ζ
1	Intr	oduction	1
	1.1	Aviation	2
		1.1.1 Flight Phases	2
		1.1.2 Flight Trajectory Data	3
	1.2	Machine Learning	4
		1.2.1 Clustering \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots	4
		1.2.2 Density-based Clustering	6
		1.2.3 Long Short-Term Memory (LSTM)	6
2	Rela	ted Work	9
	2.1	Flight phase estimation with fuzzy logic 12	1
	2.2	Other machine learning models	3
3	Met	hods 15	5
	3.1	Data	6
		3.1.1 Broadcasted Trajectory Data 16	6
		3.1.2 Simulation Data \ldots 19	9
	3.2	The classifier $\ldots \ldots 23$	3
		3.2.1 Segmentation $\ldots \ldots 23$	3
		3.2.2 Features	5
		3.2.3 Classification	6
4	Res	llts 29	9
	4.1	Simulation Data	0
		4.1.1 Different data cuts	0
		4.1.2 Different penalty influence	2
		4.1.3 Segmentation 33	3
	4.2	Broadcasted Trajectory Data	5
		4.2.1 Comparison with fuzzy logic model	9

5	Con	clusion	43		
0	5.1	Benefits of the model	43		
	5.2	Limitations of the model	44		
	5.3	Future Work	46		
		5.3.1 Supplementary Real Flight Data	46		
		5.3.2 Feature Improvement	46		
	5.4	Applications in Other Fields of Research	47		
A	crony	Yms	49		
Bi	3. Sibliography 53				

List of Figures

$\begin{array}{c} 1.1 \\ 1.2 \end{array}$	Flight phase progression over time relative to altitude	$\frac{2}{6}$
2.12.22.3	Membership functions used for flight phase estimation by Sun et al. [41]	11 11 12
3.1 3.2 3.3	Structure of the model with different data sources	15 17 24
4.14.24.3	Validation accuracies, loss, and loss penalty influence during train- ing of 3500 epochs with penalty ($\alpha = 4$) and without Results of the model with K-means segmentation and loss penalty Classification performance on a noisy flight: comparison between fuzzy logic [41] and this work	33 37 41
5.1	Comparison between computed rate of climb and original rate of climb variable.	47

List of Tables

2.1	Comparison of variables used and phases identified by related work and this work	10
3.1	Quality Statement of ADS-B flights.	18
3.2	Flight phases used in this work combined as compared to ICAO	20
3.4	The rules applied for the rule based flight phase identification from	20
	aircraft variables.	22
3.3	Aircraft state variables used in this work to label the data	22
3.5	The average time, number of segments and time per segment of a	
	flight phase in a full-length flight (flown in simulator)	24
4.1	Influence of different data clippings on validation accuracy and test	
	accuracy	31
4.2	Influence of penalty on loss on identification performance	32
4.3	K-means segmentation compared to uniform segmentation on the	
	simulation test data.	35
4.4	The average time of a flight phase over different data clippings	36
4.5	The models' performance on ADS-B test data	38
4.6	Comparison of model in this work to the fuzzy logic approach by	
	Sun et al. $[41]$	40

Chapter 1 Introduction

Flight phases are part of the aircrafts state and essential for many applications of aviation. Air traffic management uses flight phases in trajectory prediction [23], which is essential for traffic flow prediction at airports and accident avoidance [39]. In aviation research, flight phases are, among other purposes, used to analyse and reduce environmental impact [4, 10, 3]. The flight phase variable is either part of, or computed on the aircrafts internal data which is very sensitive and has scarce availability to the public. This is not the case for aircraft trajectory data. Sun et al. [41] present a model for the identification of flight phases on large scale with only trajectory data. This work uses internal aircraft data to label trajectory data and develop a model that identifies flight phases with supervised learning, to answer the following research question:

How much can the identification of flight phases using machine learning on aircraft trajectory data be improved by supervised learning?

The identification consists of the following steps:

- 1. Data preprocessing
- 2. Data segmentation (K-means [28])
- 3. Segment feature extraction
- Flight phase identification on segments (Long Short-Term Memory (LSTM) [17])
- 5. Transfer flight phases from segments to original preprocessed data.

In this chapter, a brief description of flight phases is given together with an introduction to using Automatic Dependent Surveillance Broadcast (ADS-B) trajectory data and why it is important for the identification of these, after which a background on the machine learning methods used in this work is provided. In chapter 2 state-of-the-art flight phase identification from ADS-B trajectory data based on fuzzy logic by Sun et al. [41] and other related work is presented. Afterwards, chapter 3 presents the details of this work's model and chapter 4 discusses



Figure 1.1: Flight phase progression over time relative to altitude.

the obtained results. Finally, the contributions of the model to aviation and machine learning research are evaluated and the possibilities of potential future work are discussed.

1.1 Aviation

Each aircraft produces a lot of data during a flight, the majority of this data stays within the aircraft during the flight and is later collected for post analysis by the airline and the manufacturer of the aircraft. In this work, this data is referred to as aircraft internal data and contains information regarding the flight phases. However, in order to manage air traffic most aircrafts transmit information regarding their trajectory, this data is received by other aircrafts but also by receivers on ground making this data public. The internal aircraft data on the other hand contains very sensitive information¹ and is thus maintained private. This is why research is aiming at enriching trajectory data [40] to allow for more data intensive processing in the field of aviation, flight phases are at the basis of this as many applications rely on them [43, 34, 49].

1.1.1 Flight Phases

Flight phases (also referred to as Phase of Flight or PoF) identify the operational stages during an aircraft's flight and are defined on board the aircraft and included in the aircraft's internal data. While there are multiple definitions and acronyms for flight phases, the most commonly used are given by International Civil Aviation Organisation (ICAO)'s Accident Data Reporting (ADREP) and International Air Transport Association (IATA)² [46] rather similar to each other, except for the

 $^{^{1} \}rm https://www.easa.europa.eu/domains/safety-management/safety-promotion/europe an-operators-flight-data-monitoring-eofdm-forum accessed on: <math display="inline">15/03/2021$

²https://www.skybrary.aero/index.php/Flight_Phase_Taxonomy accessed on: 15/03/2021

standing phase, which is not of interest to this work. This work focuses on the definitions of the ICAO standard that provide a set of constraints on aircraft parameters for each primary-phase and their sub-phases, the specifics of these constraints and parameters, however, depend on the individual aircraft. By using the ICAO standard, it will be possible to align this work with maintenance issues and accident databases in future applications. As flight phases are used in post-analysis, different models have been developed to estimate the flight phases based on aircraft data or create simulations that include flight phases. These models often generalise or group flight phases [41, 37, 44, 19, 23], as is done as well in this work. Figure 1.1 shows a schematic overview of the progression of the flight phases used in this work over time and relative to the aircrafts. A more technical definition of the flight phases used in this work and their exact relation to the ICAO standard will be given in chapter 3.

Applications of flight phases can be grouped in two main approaches: applications that focus on specific segments of the flight that correspond to one or more flight phases and the analysis of the change to a parameter or behaviour during the flight, comparing the different phases. The analysis of the environmental impact of air traffic uses such segmentation and analyses mainly two sources: sound emission and fuel emission. Sound emission analysis requires a focus on the initial and final phases as they are the ones closest to the earth [50]. Fuel emission analysis, on the other hand, excludes these [15]. Flight segmentation is also necessary for air traffic management purposes: aircraft initial mass estimation models either use the climb phase [2] or the take-off phase [42], a more recent approach [43] instead combines estimations on the different phases. As flight phases are characterised by different events or parameter values, the comparison of the aircraft's behaviour during different phases is not only done for mass estimation but also for the analysis of the behaviour of pilots and crew during the flight or for accident analysis and avoidance [1].

1.1.2 Flight Trajectory Data

The vast majority of aircrafts broadcast information about their trajectories, which contain information to determine their current position and path. This information is transmitted by means of ADS-B technology and received by ADS-B receivers around the world. Currently in Europe over 82% of the aircrafts are equipped with these transmitters and this number is expected to rise above 95% by 2025³. The OpenSky Network [38] is one of the platforms using the ADS-B technology to openly provide air traffic data for research purposes. This makes this data very accessible and widely used. With ADS-B data the aircraft's trajectory information (position, altitude, velocity, and rate of climb), obtained from the different systems inside the aircraft, is transmitted mostly every second together with the information needed to identify the aircraft and the transmission.

OpenSky collects data from various receivers worldwide and groups them into

³https://ads-b-europe.eu/ accessed on: 15/03/2021

flights, as the signals acquired by the ADS-B receivers are obtained by multiple aircrafts broadcasting their information, there can be interference between aircrafts. This, combined with the internal errors and different sampling rates of the sensors causes gaps in the data and noisy values (see figure 3.2). The quality of the data thus varies from one flight to another, therefor, in this work, a quality statement will be made regarding each analysed flight. Initial data preprocessing is thus necessary to ensure that the quality of the input data is as high as possible. More details on the processing of the data are provided in chapter 3.

1.2 Machine Learning

Machine learning algorithms are algorithms that learn from experience [29]. In this study, two of the main categories of machine learning are used: supervised learning algorithms and unsupervised learning algorithms. Supervised learning is the machine learning task of learning by example: finding a mapping between input and output by training on supervised data. In this work the input is the flight data and the output is the flight phase. Supervised data consists of two components for each element of the dataset: the input (x_i) and the corresponding labels (y_i) , where a label identifies the desired output that the model should produce from the input. Different tasks can be performed with supervised learning. In this work, the focus is on a classification task, as such the labels of the inputs consist of the class this input belongs to (the flight phase). Once the supervised model is trained, it is able to produce the correct output for new unlabeled inputs. A generalised supervised learning algorithm can be seen as a black box that approximates a function:

$$G: X \to Y$$

where X and Y are the input and output space respectively. The function is approximated by learning on a training set:

$$T = \{(x_0, y_0), ..., (x_n, y_n)\}\$$

The training set used for flight phase identification model consists of 256 flights each divided into 160 segments where each segment is represented by 9 features and is given a label corresponding to one of the 8 flight phases considered in this work. As such X is a $256 \times 160 \times 9$ matrix and Y a 256×160 matrix. The LSTM is a supervised learning algorithm that will be described in more detail later in this chapter. For unsupervised learning, the data is not enriched with a correct label. As such unsupervised learning algorithms are used to find patterns or structures in the data [5]. One of the main tasks suitable for unsupervised learning is clustering, which is used in this work and described as follows.

1.2.1 Clustering

Clustering is the concept of putting together data points that are similar and separating data points that are dissimilar. The similarity measure between data points and the representation of a cluster are the key differences between the various clustering algorithms. In this work, two different clustering algorithms are used: K-means [28] and Density-based Spatial Clustering of Applications with Noise (DBSCAN) [14]. K-means is used to group together flight data points that are similar into one segment, as similar data points belong to the same flight phase. DBSCAN, on the other hand, is used in the last step of outlier removal, by grouping the data points belonging to the flight together and removing those that do not (the outliers).

K-means

K-means [28] is a centroid-based clustering algorithm, meaning that a cluster is represented by the centroid of the data points belonging to it. A centroid is identified with a distance similarity measure, in the majority of cases squared Euclidean distance.

The standard K-means algorithm starts by randomly assigning each data point to one of the K clusters, where K is a predetermined number. It then proceeds by alternating the following two steps until convergence:

• Assigning each data point (x) to the cluster (C) with the nearest mean (μ) .

$$\forall i \in \{1, ..., k\}. \ C_i = \{\forall x \in data : ||x - \mu_i||^2 \le ||x - \mu_j||^2 \ \forall j \in \{1, ..., k\} \land j \neq i\}$$
(1.1)

• Update the clusters' means (μ) based on the newly assigned data points that compose them.

$$\forall i \in \{1, ..., k\}. \ \mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j \tag{1.2}$$

The algorithm converges when no points are assigned to different clusters, in comparison to the previous iteration or until the predetermined maximum number of iterations is reached.

In this work K-means is used for dividing a signal into K segments of variable length, where K = 160. The data corresponds to a full flight and each data point represents the speed and altitude of one second during the flight. Compared to the standard K-means algorithm, the following differences are introduced:

- The clusters are initialised with uniform segmentation, dividing the signal into K equally long segments.
- Only the edge points of each cluster are able to change their currently assigned cluster based on the nearest mean.

More details about the algorithm used are provided in chapter 3.

1.2.2 Density-based Clustering

DBSCAN [14] is one of the most known algorithms for density-based clustering, which relies on the idea that clusters consist of data space regions of high point density. Clusters are defined with the concept of core points and reachability:

- Core points are data points that have at least the minimum amount of points to constitute a cluster within the reach of the maximum distance, where the minimum amount of points and the maximum distance are predetermined.
- A point is **reachable** from a core point if it is within the maximum distance of it.

A cluster is composed by all core points reachable from one another and non-core points reachable from these core points. In this work DBSCAN is used to cluster together all the data points belonging to the flight as these will have other flight data points in their vicinity since abrupt changes within one second are physically impossible.

1.2.3 Long Short-Term Memory (LSTM)

The main component of the model presented in this work, the classifier, is given by a Long Short-Term Memory (LSTM), a type of Artificial Neural Network (NN). An Artificial Neural Network is a computational architecture that consists of layers of nodes interconnected to each other by weighted edges, inspired by the neurons and synapses of the biological brain [16].





Structure of Artificial Neural Network (NN), Recurrent Neural Network (RNN) hidden nodes, and Long Short-Term Memory (LSTM) hidden nodes.

Three types of layers compose an Artificial Neural Network as can be seen in figure 1.2a:

- The **input** layer represents the input for which a result is to be computed. The dimensionality of the input defines how many nodes are present in this layer. Many applications require feature extraction and data scaling before the raw data can be processed by the network, in this case each feature will have its own node in the input layer which receives values between 0 and 1. In this work, the features are the first and last values of a segment for each trajectory component that has been scaled to fit the interval between 0 and 1, as will be further explained in chapter 3.
- The **output** layer contains the result of the computation, depending on the task at hand the number of nodes and the meaning of their values vary. The task addressed in this thesis is a classification task, meaning that the output layer contains a node for each class, more specifically each possible flight phase, that represents the likelihood of the input belonging to that class.
- Hidden layers are the layers between the input and output layer. The number of hidden layers together with the number of nodes in each layer define the computational capacity of the NN. A simple task can be solved with a single hidden layer, however, complex tasks require multiple of them, networks with many hidden layers are called Deep Neural Networks.

For supervised learning tasks, a NN is trained by receiving feedback on the output it produces through its computation and adjusting the weights between the nodes accordingly. The difference between the produced output and the desired output is computed with a loss function and is called the loss. The influence of the loss of a single data entry on the adjustment of the weights is regulated by the learning rate. This is generally done with backpropagation, where the loss is computed on the output layer and from that the error of each node in the network is traced back by using the current weights, these same weights are then adjusted to minimize the node error.

There are many types of NNs, the one used in the model presented in this thesis is a LSTM, a type of Recurrent Neural Network (RNN) [36]. RNNs have the peculiarity of having a memory of the past inputs allowing them to handle sequential data. When feeding a sequence of data to a RNN, the output of each element of the sequence is computed not only on the current element but also through the memory of the previous elements of the sequence. As each node passes its computed value of the element (t - 1) in the sequence to the computation of that same node on the next element (t) in the sequence, this value is stored in the cell state, see figure 1.2b. A known problem of RNNs is their vanishing gradient, this occurs as the network only receives feedback on the produced outputs once the whole sequence has been processed. Once the feedback reaches the early elements of the sequence, the influence is too small compared to the later elements as the backpropagation has to unravel for each element in the sequence, making it hard to learn long-term dependencies.

The LSTM tackles this problem by introducing a long-term memory, the memory of an LSTM cell remains unchanged if not by changes applied through the forget gate activation vector and the output gate activation vector, the two leftmost vertical lines in the LSTM cell (figure 1.2c). The result of each cell at each element of the sequence is a weighted computation between the memory and the direct input. In this work, the network receives a sequence of flight segments that compose a full flight, which allows the network to learn dependencies over the full flight, and returns the flight phases for each segment of the flight after having seen a full flight.

Chapter 2 Related Work

Flight phases are essential for many applications of aircraft research and air traffic management. At the same time, they are defined on board the aircraft by the pilot or by internal system parameters [9], which are mostly unavailable on a large scale because of their confidentiality. This has called for the need to identify them externally either online or offline. Recent research shows that machine learning approaches outperform previous statistical modelling approaches [23]. However, the number of phases identified is often reduced compared to the number of existing phases, especially neglecting the takeoff, initial climb, approach and landing phases where most accidents occur [1] and which are thus much needed for air traffic management. The aim of this work is to increase the number of identified phases, including takeoff and landing, while keeping a similar accuracy to state-of-theart flight phase identification from trajectory data. It is worth mentioning that compared to other identifiers in this work, the climb and descent are not merely given by a positive or negative rate of climb as is mostly done for simplification in other models.

Tables 2.1a and 2.1b provide an overview of the different aircraft variables used and the flight phases detected by the models referenced in this chapter and this work. More information regarding the referenced models is provided as follows and more details regarding the variables and phases used in this work are found in chapter 3.

state variable	Sun et al. $[41]$	Liu et al. $[26]$	Kovarik et al. [23]	this work
			Paglione et al. [32]	
Altitude	Х	Х	X	Х
Speed	X			Х
Rate of Climb	Х	Х		Х*
XY coordinates			Х	
Pitch, Roll and True Heading angle		Х		
Power Lever Angle		Х		
Engine Fan Speed		Х		

(a) Trajectory variables used in related work

 $\ast~$ Rate of Climb is computed from altitude

flight phase	Sun et al. [41]	Liu et al. [26]	Kovarik et al. [23] Paglione et al. [32]	this work
ground / taxi	Х	Х	Х	Х
take-off				Х
climb	Х	Х	Х	X*
cruise	Х	X**	X**	Х
level	Х	X**	X**	*
descent	Х	Х	Х	Х*
turn			X***	
landing				Х

(b) Flight phases identified by related work

 $^{\ast}~$ climb is further divided into initial climb and climb, descent is divided into descent and approach, level is considered part of climb or descent

 ** cruise and level are considered the same phase

*** In a separate model than for the other phases

Table 2.1: Comparison of variables used and phases identified by related work and this work. The Sun et al. [41] model is based on fuzzy logic. Liu et al. [26] use a Gaussian Mixture Model. Kovarik et al. [23] compare 3 different models using Support Vector Machine, Long Short-Term Memory, and Neural Ordinary Differential Equations. Paglione et al. [32] use linear regression.

2.1 Flight phase estimation with fuzzy logic

An approach to the identification of flight phases from ADS-B data has been provided by Sun et al.[41] based on Density-based Spatial Clustering of Applications with Noise (DBSCAN) clustering[14] and fuzzy logic[48]. This model uses data from a single ADS-B receiver, which means it has to process all raw broadcasted messages and divide them into flight segments. This, together with denoising, is done by standardization and clustering using DBSCAN. From these single flights, it then estimates flight phases by applying fuzzy logic to flight segments of a fixed number of seconds. Fuzzy logic introduces partial truth to Boolean logic: rather than a statement being true or false, it can range anywhere in between these two values. The truth value is called the degree of membership or membership value of the input, which is calculated through the membership function. The trajectory variables used by Sun et al. are: altitude (H), speed (V), and rate of climb (RoC). Values of each of these variables are mapped to a membership value with the membership functions in figure 2.1.



Figure 2.1: Membership functions used for flight phase estimation by Sun et al. [41]. The upper three graphs show the membership functions for the region of values of each flight trajectory variable and the lower graph shows the membership function for each phase given the logic statement that defines it.



Figure 2.2: Valid transitions between flight phases according to Sun et al. [41]: ground (GND), climb (CL), cruising (CR), descending (DE), and levelling (LVL).



(a) Flight with valid phase transitions.

(b) Flight with invalid phase transitions.

Figure 2.3: Flight phases estimated as by Sun et al. [41] on sample data provided. Possible phases are ground (GND), climb (CL), cruising (CR), descending (DE), levelling (LVL) or unknown (NA). The flight labeled with invalid transitions shows a transition from climb to descent and from descent to cruise, both not valid.

The following logic is applied to define the flight phases are defined:

- $Ground = H_{gnd} \wedge V_{lo} \wedge RoC_0$
- $Climb = H_{lo} \wedge V_{mid} \wedge RoC_+$
- $Cruise = H_{hi} \wedge V_{hi} \wedge RoC_0$
- $Descent = H_{lo} \wedge V_{mid} \wedge RoC_{-}$
- Level $flight = H_{lo} \wedge V_{mid} \wedge RoC_0$

To evaluate this model, Sun et al. count the number of invalid transitions, where the valid transitions are the ones shown in figure 2.2. One could argue that the image shows possible transitions from cruise to climb and from descent to climb, which are not universally identified as possible transitions [9]. Of the 500 flights used for validation, less than 5% had at least one invalid transition as defined by the authors. The main disadvantage of this tool is that the number of flight phases identified is rather small and with the available data, the takeoff and landing phase could be identifiable using acceleration [41]. Figure 2.3 shows a positive and negative example of the identification with the model obtained by execution of the model and data given in their public repository¹.

¹https://github.com/junzis/flight-data-processor accessed on: 15/03/2021

2.2 Other machine learning models

A different model for flight phase estimation with machine learning has been presented by Liu et al., using a Gaussian Mixture Model (GMM) clustering [26]. This unsupervised model offers nearly the same phases as the previously discussed model by Sun et al. but rather than trajectory data, this model also uses aircraft internal data such as: pitch angle, roll angle, true heading angle, power lever angle, and engine fan speed.

There are also models that aim at predicting the phase of flight as part of trajectory prediction. Kovarik et al. [23] compared 3 machine learning models for this purpose to a simple regression model proposed by Paglione et al. [32]: Support Vector Machine (SVM), Long Short-Term Memory (LSTM), and Neural Ordinary Differential Equations (NODE). They found that the LSTM model performed best among the 3 analysed. The LSTM model consists of two separate networks that predict the next horizontal or vertical flight phase one step ahead. The horizontal flight phases consist of straight and turn, however, the vertical flight phases consist of ascending, descending, and level flights. These flight phases were predicted from X and Y coordinates and altitude but do not specify the optimal window length and or sequence length used by the LSTM.

The model in this work has combined the approach of clustering and LSTM. Firstly, similar values are grouped together, as similar values often belong to the same flight phase. In contrast to previous work, more clusters than the number of flight phases are identified and kept continuous in time to give them to a LSTM that learns the sequential dependencies of these clusters and identifies flight phases with potentially multiple clusters. The details of this implementation are provided in the next chapter.

Chapter 3

Methods

In this chapter the details of the implementation of the model and the data used are described, a schematic overview can be seen in figure 3.1. First, the their usage and labeling of the different sources of data are described. After which, the various components of the model are described: segmentation, feature extraction, and classification.



Figure 3.1: Structure of the model with different data sources, which require different preprocessing. ADS-B data contains noise originating from transmission interference, different sampling rate of the sensors, and internal sensor errors. The ADS-B data thus requires cleaning, which is done in the preprocessing. Simulation data requires clipping in order to resemble the real data more closely, since the data collected from flights is often incomplete.

3.1 Data

As internal aircraft data is scarce in availability, but required for the labeling of the data, the model presented is trained on simulation data and evaluated both on the simulation data itself and on the 7 real flights, of which both the broadcasted trajectory data and the internal data, are available. These two sources differ in their form of acquisition and thus require different preprocessing: the Automatic Dependent Surveillance Broadcast (ADS-B) data mostly requires cleaning (see subsection 3.1.1) and the simulation data clipping (see subsection 3.1.2) in order to resemble the real data more closely.

3.1.1 Broadcasted Trajectory Data

ADS-B was developed as surveillance technology, able to provide easy access to flight tracking and planning¹, at the same time . For this purpose aircrafts broadcast a message containing the message identification, flight identification (call sign), GPS-derived latitude and longitude, barometric altitude, rate of climb, azimuth direction and speed every one or two seconds. All flights considered in this work had 1 Hz transmissions, which is the most common transmission rate. The signals are then acquired by the ADS-B receivers, which receive signals of multiple aircrafts broadcasting their trajectory on the same channel. As such, there is some interference between transmissions from different aircrafts [30]. This combined with the internal errors from the sensors and sometimes different sampling rates is the cause of gaps in the data and noise (see figure 3.2) [35].

Initial data preprocessing is thus necessary to ensure that the quality of the input data is as high as possible. For this, outliers are removed, using filters and density based clustering, after which the data is interpolated. The details of this preprocessing are described in the following subsection. As the quality of the data varies between flights, the data preprocessing also provides a quality statement for each flight, airport and route. The quality statement can be used to find a relation between the accuracy of the flight phase identification and the quality of the data.

Broadcast Data Preprocessing

As will be subsequently explained in this chapter, the ADS-B information of interest to the model are the altitude and speed of the aircraft at each time point. The following preprocessing steps are, hence, applied to these two variables.

At first, the invalid values in the data are removed. When a new value for a variable is missing, the old one is repeated in the new transmission, these repeated values are invalid data points. As such, the values that are identical to their predecessors are removed. Since the values are given up to the 10th decimal in speed and 2nd decimal for altitude, the event of two subsequent measurements producing the identical value is excluded. For the values that have valid predecessors, outliers are removed based on a physically possible change within one second, the

¹https://ads-b-europe.eu/ accessed on: 15/03/2021

transmission rate of the data of interest. For altitude this is 150 meters, while for speed, 1.5 meters per second. This first step eliminates the majority of outliers and is sufficient for the speed variable. However, the altitude variable is often more noisy and in order to provide a more reliable interpolation, the second and third steps further eliminate the remaining outliers.

The second step consists in applying the same thresholds as the first but this time to the median of the 12 points before and after the point taken into consideration. This allows to find outliers also where there are invalid points in the vicinity of the outlier, or when there are multiple consecutive outliers. This method is applied to the altitude variable since there are no valid sudden variations in it which, instead, could occurs in the speed variable.

The third step consists in applying a density-based scanner (Density-based Spatial Clustering of Applications with Noise (DBSCAN)) aimed at identifying the main flight as a single cluster. This is done by providing relatively large hyperparameters. The minimum points to form a cluster is 75 and the maximum distance between points is 300, based on Euclidean distance and calculated over the 2 dimensions of time and altitude. At this point, less than 1% of flights have visibly remaining outliers and the missing values are linearly interpolated between two valid values.

Finally, the edges where either of the two variables presents missing data are removed and the flights converted to a different metric system. For this application, the Deutsches Zentrum für Luft- und Raumfahrt (DLR) prefers the use of feet (ft) rather than meters (m) and knots (kts) rather than meters per second.(m/s).



Figure 3.2: Comparison between original noisy data in red and preprocessed in blue. The original data has 30% missing values and 14% outliers, which are identified and interpolated.

The preprocessing of steps for ADS-B data are defined as follows:

1. Removing values where difference within one second is larger than physically possible.

$$alt(t) = \begin{cases} NaN, & \text{if } alt(t-1) \neq NaN \land abs(alt(t) - alt(t-1)) > 150 \, m \\ alt(t), & \text{otherwise} \end{cases}$$
(3.1)

$$spd(t) = \begin{cases} NaN, & \text{if } spd(t-1) \neq NaN \land abs(spd(t) - spd(t-1)) > 1.5 \, m/s \\ spd(t), & \text{otherwise} \end{cases}$$
(3.2)

2. Removing values where distance to the median over a window of empirically found length is greater than the same threshold of the previous point.

$$alt(t) = \begin{cases} \text{NaN}, & \text{if } 12 \le t \le max(t) - 12 \land \\ & \land abs(alt(t) - med(alt(t - 12, ..., t + 12))) > 150 m \\ alt(t), & \text{otherwise} \end{cases}$$
(3.3)

- 3. Applying a density based clustering algorithm that identifies the main group of data points and those outside it.
- 4. Linear interpolation for all points between two valid points.
- 5. Cutting off the edges where either of the used values presents invalid points.

This procedure was applied to 2088 flights, of which the summary of the quality statement is reported in table 3.1. 7 of these flights had accompanying sensor data that allows to label them.

The labeling is performed with the tool provided by the DLR described in subsection 3.1.2. This tool was developed for the real internal aircraft sensor data of the DLR's research aircraft and with some minor adjustments extended to provide the same labelling for the simulation data. Once the internal sensor data is labeled, the two sources representing the same flight, the broadcasted trajectory and the internal data, are aligned. At which point the labels are transferred from the internal flight data to the ADS-B data.

	Average	Min	Max
Percentage of missing values	33%	20%	77%
Percentage of outliers	13%	0%	22%

Table 3.1: Quality Statement of 2088 ADS-B flights analysed.

3.1.2 Simulation Data

As the number of available ADS-B flights with sensor data is limited, the X-plane² flight simulator is used to produce additional sensor data which will be labeled with the flight phase tool. In the X-plane flight simulator, an aircraft model can be chosen and flown, the creators of this simulator claim that with the advanced computation of aircraft forces, their flight model is much more detailed than what is used by most other flight simulators. In fact the X-plane simulator software can be used to create Federal Aviation Administration of the United States Department of Transportation (FAA) certified³⁴ training devices, which require detailed and close to reality simulations. FAA approved training devices can, for example, be used for pilot training.

For the collection of simulation data in this work, a Boeing 737, one of the most commonly used commercial aircrafts⁵, was flown by the artificial intelligence pilot made available by the simulator. These flights have no corresponding ADS-B data, however, the simulator provides the same trajectory information as ADS-B and will thus be used for training the model. Although the two sources provide the same values, there are differences that cause a decrease in performance on the real data compared to the simulation data. The main difference is that the simulation provides much smoother and clean data than the transmitted ADS-B data. This is mainly due to the noise of the transmission itself but also because the simulator flies without air traffic which can effect the maneuvers of the aircraft. More details regarding the differences in the sources of data will be discussed in the following chapters.

A total of 421 simulation flights were recorded, of which 256 were used for training, 65 for validation and 100 for testing. Each of these flights contains the variables necessary to label the data and the trajectory information needed by the model.

Simulation Data Preprocessing

The simulation data includes the values of the aircraft state variables that can be seen in table 3.3. The majority of the variables are solely needed for labeling the data. As there is no unified or universal definition to quantitatively specify flight phases, many phases of flight are descriptively defined without quantitative constraints on the aircraft state variables [12]. In this research the International Civil Aviation Organisation (ICAO) standard is used and the definitions are translated into specific aircraft state variables rules. The relation between the complete set of ICAO Accident Data Reporting (ADREP) primary phases and sub-phases and the phases used in this work can be seen in table 3.2.

²https://www.x-plane.com/

³https://www.x-plane.com/pro/certified/ accessed on: 15/03/2021

⁴https://www.faa.gov/regulations_policies/advisory_circulars/index.cfm/go/do cument.information/documentID/1034348 accessed on: 15/03/2021

⁵https://centreforaviation.com/analysis/reports/aircraft-fleets-western-v-eas terncentral-europe-airbus-leads-orders-410122 accessed on: 15/03/2021

The Standing phase is excluded because there are no ADS-B transmissions in this phase. The Maneuvring primary phase and Holding sub-phase are excluded as they are phases that do not occur in commercial aircrafts. The taxi, cruise, approach, and landing phases are a combination of their sub-phases for simplicity. The Go Around and Rejected Takeoff are infrequent sub-phases, which is why they are grouped together with their more frequent counterparts.

ICAO primary phases	ICAO sub-phases	Phase in this work
Standing (STD)	-	-
	Taxi to Runway	
Taxi (TXI)	Taxi to Takeoff Position	taxi (TXI)
	Taxi from Runway	
Takooff (TOF)	Takeoff	take off (TOF)
	Rejected Takeoff	
Initial Climb (ICL)	-	initial climb (ICL)
	Climb to Cruise	climb (CLI)
	Cruise	cruiso (CBZ)
En Route (ENR)	Change of Cruise Level	
	Descent	descent (DST)
	Holding	-
Maneuvering (MNV)	Aerobatics	
	Low Flying	_
	Initial Approach (IFR)	
Approach (APR)	Circuit Pattern - Downwind	
Visual Flight Rules (VFR)	Circuit Pattern - Base	approach (APR)
	Circuit Pattern - Final	
	Circuit Pattern - Crosswind	
	Go Around	
Landing (LDG)	Flare	landing (LDC)
	Landing Roll	

Table 3.2: Flight phases used in this work combined as compared to ICAO ADREP. The standing and maneuvring phases, together with the holding sub-phase, are excluded for being out of the scope of this work. The taxi, approach, and landing are a combination of their sub-phases for simplicity, with the exception of the go around sub-phase which together with the rejected takeoff sub-phase are included in their primary phase because of their infrequency.

The data labeling consists of applying a rule-based approach for flight phase identification based on the ICAO standard. For this purpose, a tool has been devel-

oped by the DLR⁶ to augment aircraft internal sensor data with the corresponding flight phase as closely to the ICAO definitions as possible. The relation between the rules used in this work and the ICAO nomenclature can be found in table 3.2. This tool has been developed for an Airbus A320 aircraft⁷ for which the DLR has access to some internal flight data (the 7 real flights used in this work). This tool has been adapted to the X-plane simulation data of the Boeing 737⁸ as the state variables for the two sources of data slightly differ. Table 3.3 shows the adaptation made on the simulation data to fit the real data state variables. The specific rules for each phase in this work are described in table 3.4 with the use of the abbreviations given in table 3.3. In these tables, the altitude indicates altitude above runway, the functions max(param), min(param), abs(param) indicate the maximum, minimum or absolute value of that parameter and ts(event) indicates the time frame (in seconds) of a certain event.

Descriptively, the flight phases of tables 3.2 and 3.4 are defined as follows:

- taxi (TXI): The phase before take-off or after landing when the engine is on and the aircraft is on ground.
- take-off (TOF): The engine is at at least 80% of its maximum achieved during the flight and the aircraft is below 35 feet above the runway.
- initial climb (ICL): From the end of the take-off phase until the altitude reaches 1000 feet above the runway.
- climb (CLI): From the end of the initial climb phase until the cruise phase.
- cruise (CRZ): The altitude rate is near to 0 and the aircraft's altitude is above 1000 feet altitude and one of the following conditions is true: the aircraft is flying at near to maximum altitude or the altitude rate is near to 0 for more than 6 minutes.
- descent (DST): The aircraft's rate of climb is negative for more than 2 minutes and the approach phase has not yet started.
- approach (APR): The aircraft's rate of climb is negative and the altitude is below 1000 feet from the runway.
- landing (LDG): 5 seconds before the aircraft reaches the ground and until it leaves the runway (i.e., when it starts steering) or stops.

⁶developed by Alexander Kamtsiuris (alexander.kamtsiuris@dlr.de) in the department of Process Optimisation and Digitalisation, part of the Maintenance Repair and Overhaul institute.

⁷https://www.airbus.com/aircraft/passenger-aircraft/a320-family.html accessed on: 15/03/2021

⁸https://www.boeing.com/history/products/737-classic.page accessed on: 15/03/2021

Flight Phase	Rule
TXI	$gear_compr \land tgt_epr > 0 \land (ts < ts(TOF) \lor ts > ts(LDG))$
TOF	$alt > 35 \land gear_lvr_down \land cmd_epr \cdot tgt_epr > 0.8 \cdot max(tgt_epr)$
ICL	$ts > ts(TOF) \land 35 < alt < 1000$
CLI	$ts > ts(ICL) \land ts < ts(CRZ)$
CRZ	$-500 < roc < 500 \land$
	$\wedge \left((alt > max(alt) - 1000) \lor (alt > 1000 \land (ts_{end} - ts_{begin}) > 360) \right)$
DST	$roc < -10 \land (ts_{begin} - ts_{end}) > 120 \land ts < ts(APR)$
APR	$roc < -10 \wedge alt < 1000$
LDG	$ts > ts(TOF) \land ts_{begin} - ts(gear_compr) < 5 \land$
	$\land (abs(steer_ang) < 3 \lor spd == 0)$

Table 3.4: The rules applied for the rule based flight phase identification from aircraft variables.

State Variable	Abbreviation	Measurement Unit	Labeling (L) / Conversion (C)	Conversion	
Time	ts	seconds	L	-	
Barometric altitude	alt	feet	L	-	
Ground Speed	spd	knots	L	-	
Altitude Rate	roc	feet per minute	L	$\frac{alt}{ts}$	
Target Take-off	tot opr		Т		
Engine Pressure Ratio	tgt_epr -		L	-	
Throttle	thro	-	С	-	
Commanded Take-off	and opr		I.	$1 + thro \cdot (mar(tat orr) - 1)$	
Engine Pressure Ratio	cinq_epi	-	L	$1 + inio \cdot (max(igi_epi) - 1)$	
Landing Gear Deployment	gear_lvr_down	boolean value	L		
Engine Normal Force	norm	pound	С		
Force on Main Landing Gear	gear_compr	boolean value	L	$\begin{cases} True, & if norm > 0\\ False, & if norm \le 0 \end{cases}$	
Nose Gear Steering Angle	steer_ang	degrees	L	$\begin{cases} steer_ang, & if gear_comp \\ 0, & if \neg gear_comp \end{cases}$	

Table 3.3: Aircraft state variables used in this work to label the data: internal aircraft values used for labeling (L) and the internal aircraft values used to convert or compute (C) differing or missing values in the simulation data.

As ground data is not often available in ADS-B due to the complexity of transmission such as obstacles, the flight data often does not include, or only partially includes, phases that are close to the ground. These are the taxi, take-off, initial climb, approach, and landing phases. For this purpose, the training and validation set of the simulation data has been replicated with different clippings in the beginning and in the end to more closely resemble the ADS-B data. Each flight can thus appear in the following forms:

- Initial cut: an arbitrary point is chosen between the beginning of the take-off phase and the end of the initial climb where the flight starts.
- Final cut: an arbitrary point is chosen between the beginning of the approach phase and the end of the landing phase at which the flight ends.
- Dual cut: a flight has both an initial cut and a final cut.
- No cut: a flight is left intact.

In chapter 4 the effects of different combinations of cuts will be evaluated on the validation set consisting of an equal distribution of these cuts and the real data.

3.2 The classifier

The model developed for the flight phase identification consists of two main steps: the segmentation of the flight and the classification of the segments, after each segment is translated into features. For the segmentation a variation of the Kmeans clustering algorithm [28] is used, while for the classification a Long Short-Term Memory (LSTM) [17] is used with a loss penalty function.

3.2.1 Segmentation

The first step of the model is the segmentation of the flight, which consists of dividing it into a fixed number of segments. This is achieved by using a variation of the K-means algorithm, further referred to as K-means segmentation.

K-means segmentation initializes segments by dividing the input into equal parts after which it allows the edge points of segments to either belong to their current segment or the neighboring one, based on their distance to the segment means and if their current cluster has at least 4 points belonging to it. When two neighboring edges both try to change their cluster of belonging, only the one that has a bigger difference in distance between the two means is allowed to do so.

The hyperparameters of this algorithm are the number of clusters, the maximum number of iterations, and the weights used in the distance function:

- $n_{clusters}$ is found empirically with the right trade-off between size and the minimum error introduced, it is set to 160. The mean error introduced by the segmentation for the different values can be seen in figure 3.3
- The distance weights have been found with a parameter gridsearch. 10 weights in the range (0,1] are considered for each of the trajectory variables. The optimal weights found for the minimum error introduced by segmentation were 0.1 altitude and 0.8 for speed.
- The maximum number of iterations is set by taking the 95th percentile of iterations until convergence which corresponds to 100 iterations.



Flight phase	time (s)	segments	segment time (s)
TXI	397	35.2	11.3
TOF	22	2.5	8.8
ICL	15	1.0	15
CLI	197	17.7	11.1
CRZ	174	13.0	13.4
DST	907	81.4	11.1
APR	70	5.8	12.1
LDG	30	3.4	8.8

Figure 3.3: Classification error introduced through segmentation for different number of clusters, the number of segments used for the model is marked in red.

Table 3.5: The average time, number of segments and time per segment of a flight phase in a full-length flight (flown in simulator).

The result is clusters that represent segments that are continuous in time, which size varies with the amount of change over time. The pseudo code can be found in algorithm 1 where:

- 1. x is the 2 dimensional input array of shape $length_{flight} \times variables_{flight}$, in this case the number of variables is 2 (altitude and speed), each normalised by variables subtracting that variable's minimum of the flight and dividing by the maximum of that variable in the flight.
- 2. c is the array of length $length_{flight}$ that indicates the cluster for each input entry.
- 3. μ is the 2 dimensional array of shape number_{clusters} × variables_{flight}
- 4. div(x, y) is the integer division function.
- 5. mod(x, y) is the modulo operation.
- 6. $get_means(x, c)$ is the function that computes the means of each cluster.
- 7. dist(x, y) is the weighted Euclidean distance between two arrays of the same shape.

The aim of this segmentation is to be able to reduce the error induced by phases overlapping in a single segment. A change of flight phase most likely occurs at a point where there is more change, i.e., where there is a bigger distance between two consecutive data points.

```
Algorithm 1 K-means segmentation
 start \leftarrow 0
for i \leftarrow 1, ..., n_{clusters} do
           end \leftarrow start + div(len_x, n_{clusters})
          if i \leq mod(len_x, n_{clusters}) then
                     end \leftarrow end + 1
          end if
          c[start:end] \leftarrow i
           start \leftarrow end
 end for
 \mu \leftarrow get\_means(x,c)
 c' \leftarrow c
for iter \leftarrow 0, ..., n_{iterations} do
          for i = 1, ..., flight_len do
                     if counts(c', c[i]) > 4 then
                               if c[i] \neq c[i+1] \land dist(x[i], \mu[c[i+1]]) < dist(x[i], \mu[c[i]]) then
                                         c'[i] \leftarrow c[i+1]
                               else if c[i] \neq c[i-1] \land dist(x[i], \mu[c[i-1]]) < dist(x[i], \mu[c[i]]) then
                                         if c'[i-1] = c[i-1] then
                                                    c'[i] \leftarrow c[i-1]
                                         else if dist(x[i-1], \mu[x[i-1]]) - dist(x[i], \mu[c[i-1]]) < dist(x[i], \mu[c[i]]) - dist(x
                                         dist(x[i], \mu[c[i-1]]) then
                                                    c'[i] \leftarrow c[i-1]
                                                    c'[i-1] \leftarrow c[i-1]
                                         end if
                               end if
                     end if
          end for
          c \leftarrow c'
           \mu \leftarrow get\_means(x,c)
end for
```

3.2.2 Features

After the flight has been divided into segments, these segments need to be given to the classification network, this is achieved by extracting features from each segment. As has been mentioned previously, the aircraft state variables given to the model are altitude and speed, together with time. Other models [41, 23] also used the rate of climb and XY coordinates. The latitude and longitude provided in ADS-B data can be transformed into XY coordinates [13]. Since both XY coordinates and latitude and longitude are referenced to the earth, for the network to be able to generalize flights from different airports, the origin coordinates are required. ADS-B flights, however, are often incomplete which means that the origin coordinates are not always available. It would be possible to retrieve the coordinates from the origin airport, but this is out of the scope of this work. To circumvent this issue, the latitude and longitude could be used taking their difference over time, yet this would correspond to using the speed, as such, these variables are excluded in this work. The rate of climb, on the other hand, is fairly easy to compute through the difference in altitude, as such, rather than using the aircraft variable, it is directly computed from the altitude and introduced as a feature, as has been done for the labelling provided by DLR.

The following features given to the network are computed for each segment provided by K-means segmentation:

- 1. length of the segment (n)
- 2. initial altitude (alt_0)
- 3. final altitude (alt_n)
- 4. initial speed (spd_0)
- 5. final speed (spd_n)
- 6. initial rate of climb $(alt_1 alt_0)$
- 7. final rate of climb $(alt_n alt_{n-1})$

Each of these features is normalised: for the first 5 features (length, altitude, and speed), this consists in dividing each of them by the maximum value of that feature in each flight. For the rate of climb features, this consists in subtracting the minimum value of that feature in the flight it belongs to and subsequently dividing it by the maximum. The altitude and speed are always positive, and if a flight is incomplete, the minimum altitude or speed might not correspond to the value of these in a complete flight.

3.2.3 Classification

The features extracted from the segments, as explained previously, are used by the classification network that labels each segment with a flight phase. For the classification, a neural network is needed that is able to capture longer temporal relations of a flight. For this purpose, an Long-Short Term Memory (LSTM) model suits this task as it was designed for this purpose [17]. The LSTM receives features of the segments, described previously, as inputs and learns to classify the clusters according to the prevailing label of that cluster. The Artificial Neural Network (NN) is implemented in Pytorch [33] and consists of an input layer, 2 layers of 16 LSTM cells, followed by an activation layer consisting of the logarithm of a softmax function. The output is a value between 0 and 1 for each of the classes, where 1 indicates the segment belongs to that class and 0 indicates that it does not belong to that class. For training, a batch size of 16 and a negative log likelihood loss (NLLL) are used for stochastic gradient descent. The overview of the architecture
of the classifier is shown in the pseudocode 2. The hyperparameters: number of layers, number of hidden units, and batch size have been found with an initial hyperparameter gridsearch. Initial evaluation of the model shows that not every flight phase is identified with the same accuracy, the shorter flight phases present more inaccuracies. There are 2 possible sources for this problem: segmentation, the shorter the flight phase, the higher the chance of possible relative inaccuracy from inaccurate segmentation, and the problem known as class imbalance. Class imbalance is a problem that emerged as machine learning evolved into an applied technology, it occurs in classification tasks when there are many more instances of some classes than others. [11]. For flight phase identification, this is due to the fact that different phases have different durations, as can be seen in table 3.5.

Class imbalance can be dealt with on three different levels: in the data itself by sampling, in the algorithm loss function, and in the feature design [27]. As the aim of the task is to capture the dependencies and relations between classes in the sequential data, resampling specific classes would change the distribution and thus disrupt the natural progression of phases. For this reason, a penalty term (equation 3.4) is added to the loss function, an approach that has been proven successful in multiple studies [8, 7, 51]. This penalty consists of the average of the false negative rate [25] and the false discovery rate [6] of each flight phase, multiplied by an influence factor α . The function *false* in the pseudocode 2 refers to wrongly identified outputs in the given segment, which is divided over the total amount of values of that phase.

$$penalty = \alpha \cdot \frac{\sum_{c \in classes} \left(\frac{FP_c}{TP_c + FP_c} + \frac{FN_c}{TP_c + FN_c}\right)}{2 \cdot ||classes||}$$
(3.4)

FP = false positive, TP = true positive, FN = false negative

In the next chapter the results of models with different penalty influences (α) and training cuts are compared, and the interpretation of the output of the classifier is described.

Algorithm 2 Classification network training

 $\begin{array}{l} \textbf{for } e \leftarrow 0, ..., n_{epochs} \ \textbf{do} \\ \textbf{for } x, y \ in \ data_{training} \ \textbf{do} \\ output \leftarrow model_{LSTM_{2\times 16}}(x) \\ output \leftarrow model_{LogSoftmax}(output) \\ loss \leftarrow NLLL(output, y) \\ \textbf{if } \alpha > 0 \ \textbf{then} \\ penalty \leftarrow 0 \\ output \leftarrow argmax(output) \\ \textbf{for } phase \leftarrow 0, ..., n_{classes} \ \textbf{do} \\ penalty \leftarrow penalty + \frac{false(output[phase])}{total(output[phase])} + \frac{false(y[phase])}{total(y[phase])} \\ \textbf{end for} \\ loss \leftarrow loss + \alpha \times \frac{penalty}{2 \cdot n_{classes}} \\ \textbf{end if} \\ model \leftarrow SGD(model, loss) \\ \textbf{end for} \\ \textbf{end for} \\ \textbf{end for} \end{array}$

Chapter 4

Results

After describing the details of the variations of the model presented in this work, the results of these are discussed as follows. Three sources of data are used for the comparison of the different models: validation data (65 simulation flights), simulation test data (100 simulation flights), and broadcasted test data (7 labeled Automatic Dependent Surveillance Broadcast (ADS-B) flights).

The metrics with which each source of data is evaluated rely on the concept of precision and recall [31]:

• The **precision** of each class is the number of correctly identified data points belonging to that class ($correct_{class}$) divided by the total number of data points identified as belonging to that class ($identified_{class}$).

$$precision_{class} = \frac{TP_{class}}{TP_{class} + FP_{class}} = \frac{correct_{class}}{identified_{class}}$$
(4.1)

• The **recall** of each class is the number of correctly identified data points belonging to that class ($correct_{class}$) divided by the total number of data points belonging to that class ($truth_{class}$).

$$recall_{class} = \frac{TP_{class}}{TP_{class} + FN_{class}} = \frac{correct_{class}}{truth_{class}}$$
(4.2)

TP = true positive, FP = false positive, FN = false negative

The evaluation on the validation data is performed on segments identified by the corresponding segmentation, where the class identified for each segment is the one that obtains the highest output from the model. This is because during training, the model only has knowledge of the segments, represented by their features. The following metrics are used for the evaluation on the validation data:

- **Overall validation accuracy** indicates the number of correctly classified segments over the total number of segments.
- Weighted validation accuracy refers to the average of the precision and recall computed on the segments for each class.

The evaluation on the test data, in contrast, is performed on each data point (second) of the flight. The phase identified for each segment consist of the most probable possible phase given the full sequence. That is under the constraint that only the following succession of phases is possible, allowing for phases to be skipped:

 $\mathrm{TXI} \rightarrow \mathrm{TOF} \rightarrow \mathrm{ICL} \rightarrow \mathrm{CLI} \rightarrow \mathrm{CRZ} \rightarrow \mathrm{DST} \rightarrow \mathrm{APR} \rightarrow \mathrm{LDG} \rightarrow \mathrm{TXI}$

The succession of phases is enforced to keep the phases continuous, since in future applications the model could be used for selecting only one or multiple consecutive phases of interest. Such selection, except for the taxi phase, should return a single flight section. The label of a segment is then applied to each data point belonging to that segment. The evaluation on the test data uses the following metrics:

- The **overall test accuracy** is given by the total amount of correctly classified seconds of flight over the length of flight.
- The **phase precision** is the average precision, computed on the flight seconds, of each phase.
- The **phase recall** is the average recall, computed on the flight seconds, of each phase.
- The weighted test accuracy average of the phase precision and phase recall.

The simulation test data consists of the replication of the test flights with all 4 possible cuts, which will be further explained in the next section.

4.1 Simulation Data

The different outcomes of the networks are obtained by training for 3500 epochs with different sets of configurations, each repeated 3 times: with and without loss function penalty, with different training sets obtained from different flight clippings, and with K-means segmentation and uniform segmentation. The total number of epochs was chosen as such to allow every configuration to have no further improvement in relevant accuracy up to the second decimal in percentage. These configurations will be evaluated on the simulation data after which the best performing models will be assessed on the ADS-B test data and compared to the model proposed by Sun et al. [41] in the next section.

4.1.1 Different data cuts

The first factor that is analysed for its impact on the performance of the model is the different training datasets used. The simulation consists of all complete flights that start with taxi, go through all flight phases, and end with taxi. This, however,

		Validation		Simulation test		
Data Cut	Penalty	accuracy (%)		r (%) accuracy (%)		
		overall	weighted	overall	precision	recall
B, E, B&E, full	no	98.00	97.00	97.19	89.87	88.94
B, E, B&E, full	yes	98.25	97.38	96.93	90.60	91.36
B, E, B&E	no	97.67	96.40	96.85	88.89	88.56
B, E, B&E	yes	97.77	96.03	96.57	88.76	89.66
B, E	no	97.31	95.93	96.26	87.51	86.89
B, E	yes	97.46	96.18	95.61	86.26	88.35

Table 4.1: Influence of different data clipping on validation accuracy and test accuracy. The model with every data cut and the loss penalty performs best. The data cut consists of taking the full data set and clipping it in the beginning (B), at the end (E), on both sides (B&E), or not at all (full). Data sets are given by multiple cuts, each a replication of all the flights clipped accordingly.

is not always the case in real flight data, as recorded flight data is often incomplete. The training data is thus replicated multiple times with different cuts in order for the model to learn that not each flight presents all phases, or presents the initial and final phases only partially.

Table 4.1 shows the comparison of the results of the models trained on different simulation data sets, characterised by the different cuts. This table shows an overall negative trend in performance the less the data gets. Every cut consists of taking the full data set and clipping it in the beginning (B), at the end (E), on both sides (B&E), or not at all (full). The beginning point is taken at random between the start of the take-off and the end of the initial climb. The ending point is taken at random between the start of the approach and the landing. The validation accuracies are averaged over all 3 repetitions of the same configuration, the test accuracy (overall accuracy for the models without penalty and weighted accuracy for the models with penalty).

When looking into the detailed evaluation of each phase, it appears that the penalty has the biggest impact on the precision of the take-off phase, on average, the model that uses the penalty increases the precision of the take-off by 9.95% compared to the model trained on the same data without penalty. The different precision and recall values for the different phases show how the shorter phases achieve lower values than the longer phases. When considering the models trained on 4 different cuts, the model without penalty has a Spearman correlation¹ value of 0.88 (p = 0.004) between length of phase and accuracy while with penalty this is reduced to 0.76 (p = 0.028).

¹Spearman correlation values lie in the range [-1, 1]

Over all, in nearly all evaluation measures, the models that perform best are the ones trained with all 4 possible cuts, in fact, further results will refer to this dataset, and also the simulation test set consists of the replication of all test flights with these 4 cuts.

4.1.2 Different penalty influence

After establishing the training dataset, given by the different cuts, the next factor to impact the results of the model is the loss penalty. Chapter 3 explains how a penalty factor is added to the loss function in order to compensate for the class imbalance that originates from the flight phases greatly differing in duration. This penalty is given the average rate of false discoveries and false negatives per phase, multiplied by an influence factor α . The values considered for this influence factor are 0, 3, 4, and 5, where 0 indicates no penalty is added to the loss function. The results reported are those of the training set with all 4 possible different clippings, however a repetition with other training sets show very similar behaviour.

Figure 4.1 shows the behaviour during training of a network training for 3500 epochs with and without penalty. The figure shows the best performing network based on validation accuracy as the network without penalty and best performing network based on weighted validation accuracy as the network with penalty. The network without penalty reaches its maximum accuracy of 98.08% at epoch 3378 and maximum weighted of 96.92% at epoch 3045. The network with penalty reaches its maximum weighted accuracy of 97.82% at epoch 3265 and its best validation accuracy of 98.27% at epoch 1840. One can note that when adding the penalty to the loss function, the weighted validation accuracy increases at a faster rate and reaches a higher maximum value, it also shows a positive impact on the overall validation accuracy as it reaches its maximum relatively early. As can be seen in figure (4.1), both models do not overfit to the data, this is most likely the case because of the complexity of the data and simplicity of the model.

	contribution to total loss	weighted validation accuracy	validation accuracy
α	(%)	(%)	(%)
0	0	97.00	98.00
3	67.55	97.25	98.05
4	73.53	97.38	98.25
5	77.44	97.31	98.19

Table 4.2: Analysis of influence of loss penalty (α) on identification performance. Both overall and weighted validation accuracy are highest with $\alpha = 4$.



Figure 4.1: Validation accuracies, loss, and loss penalty influence during training of 3500 epochs with penalty ($\alpha = 4$) and without. With penalty the model reaches a higher weighted validation accuracy and learns faster.

4.1.3 Segmentation

With the fixed training dataset and hyperparameters for the classifier, the impact of the segmentation algorithm is analysed. All previous models use the K-means segmentation that has been designed to more closely capture the boundaries between phases that supposedly correspond to moments of greater change in the data. K-means segmentation allows each data point to belong to the nearby segment with the nearest mean with regards to speed and altitude. Uniform segmentation, on the other hand, divides the flight into 160 equally long parts, thus not considering the values of speed and altitude. In this section, the models that use uniform segmentation and K-means segmentation are compared both with and without penalty and with all 4 possible cuts. Table 4.3 shows the detailed performance of these models. It is important to mention that the model with uniform segmentation had no segment length feature as this feature would carry no meaning. Considering the average of overall and weighted accuracy the uniform segmentation model with loss penalty (97.97%) performs better than the K-means segmentation model with loss penalty (97.82%) in validation. However, this is not the case for the average test accuracy (K-means=93.96%, Uniform=93.83%). This indicates that uniform segmentation allows for better training, but loses its advantage when the segment labels are applied to single data points of the flight. The source of this behaviour could be that the segment length feature causes noise to the network during training. The main advantage of the uniform segmentation lies in the weighted accuracy, which is greater in the validation performance, but still present in the test performance. The rows with phase precision and phase recall in table 4.3 allow to consider the performance on the test data in closer detail. From these results, it is apparent that the difference in weighted accuracy originates from the phase recall component of the penalty.

As has been stated in previously in chapter 3, K-means segmentation is aimed at more closely capturing the right moment of transition from one phase to another. This will have the greatest impact on the shorter phases as for these phases an overlap within a segment has a bigger impact on their overall precision and recall. The positive effect can be noticed in the precision of the short phases in the models with loss penalty but inverse in their recall: the K-means segmentation model has a greater precision on the phases with shorter segments, take-off and landing, compared to the uniform segmentation model, yet a lesser recall. In other words, the K-means segmentation model has fewer false positives but more false negatives than the uniform segmentation, this is true on average, but mostly for the short phases. The penalty model with K-means segmentation, in fact, predicts fewer data points as take-off or landing, 11 and 18 seconds on average per flight, respectively, than the uniform segmentation, which predicts 14 seconds of take-off and 20 seconds of landing on average per flight. These results lead to the hypothesis that the segment length is used for the identification of these phases, acting as a further threshold.

Table 4.3 also shows that the loss penalty appears to have a greater impact on the models that use K-means segmentation rather than those that do not. This could indicate that the penalty function increases the use of the segment length to improve the network's performance on the flight phases. In fact, the difference in weighted accuracy is much smaller in the uniform segmentation models than in the K-means segmentation models, as both achieve a better result.

In the next chapter, three of the models described will be compared on their performance on the ADS-B data: K-means segmentation with penalty, uniform segmentation with penalty, and uniform segmentation without penalty.

		K-means K-means		Uniform	Uniform
		w/o penalty	w/ penalty	w/o penalty	w/ penalty
Overall validation accuracy (%)		98	98.25	98.19	98.17
Weighted	validation accuracy (%)	97	97.38	97.68	97.77
Overall test accuracy (%)		97.19	96.93	96.8	96.58
Weighted test accuracy (%)		89.41	90.98	91.02	91.07
	TXI	98.70	98.71	99.05	99.06
	TOF	84.99	96.18	88.32	90.68
	ICL	71.67	75.99	78.70	78.71
Dracicion	CLI	96.6	96.76	95.45	96.43
(%)	CRZ	96.84	96.47	95.36	97.27
	DST	99.02	98.36	99.27	99.59
	APR	83.72	74.79	78.4	69.22
	LDG	87.41	86.36	85.93	84.96
	Average	89.87	90.60	90.06	89.49
	TXI	98.96	98.82	98.89	98.84
	TOF	69.95	71.07	86.81	85.4
	ICL	73.73	84.68	77.27	78.86
Decell	CLI	96.76	96.67	97.25	96.73
(%)	CRZ	97.56	97.57	95.79	96.72
	DST	98.60	97.33	97.39	96.53
	APR	88.86	96.03	92.72	97.09
	LDG	87.08	88.74	89.69	90.96
	Average	88.94	91.36	91.98	92.64

Table 4.3: K-means segmentation compared to uniform segmentation on the simulation test data. Uniform segmentation obtains a better performance on the validation accuracies that are measured over the segments. The K-means segmentation compensates with the overall test accuracy and the average phase precision. This means that with uniform segmentation the network trains better but that the advantage of the K-means segmentation remains when transferring the segment labels back to the single data points of the flight.

4.2 Broadcasted Trajectory Data

After the best performing models have been chosen based on their performance on the simulation data, these models' performance is analysed on the 7 flights whose data was recorded with ADS-B and that could be labeled with the internal flight records. As can be seen in table 4.5 the results on the ADS-B data are clearly not as good compared to the simulation data. While the difference in overall accuracy is less than 3% between the best performance on simulation and real data. The weighted test accuracy suffers a decrease of over 30%. There are different possible causes for this: there are several aspects that differ between the simulation and

Flight phase	Average time per flight (s)					
right phase	B, R	B, E, B&R	B, E, B&E, full	real data		
TXI	198	132	198	-		
TOF	13	11	14	1		
ICL	13	13	13	11		
CLI	197	197	197	717		
CRZ	174	174	174	1569		
DST	906	906	906	1202		
APR	56	50	59	50		
LDG	17	12	17	0		
Total	1574	1495	1578	3550		

Table 4.4: The average time of a flight phase over different data clippings. Simulation flights are much shorter (26 minutes on average) than the real flights (59 minutes on average). The cruising phase accounts for over 70% of the additional length.

the ADS-B data. The most concrete difference lies in the length of the flights, as can be seen in table 4.4, the average length of a flight is 59 minutes whereas the average length of the flights in the dataset with 4 different cuts is 26 minutes, which is rather short for a flight. This has an impact especially on the short phases since the model has a relative perception of time (the sequence length, like other features is normalised), for a longer flight the shorter flight phases (take-off, initial climb, approach, and landing) are not longer than in a shorter flight. This means that from its training a model expects to find, for instance, two take-off segments, however, if the flight is double as long as the result should only be one take-off segment. The same principle applies for all phases, however, the cruising phase is less subject to it as it is the phase with most variable length, also in the training data. The hypothesis is that this is the main cause for the underperformance of the model on the ADS-B data.

Another aspect is that the real data is less smooth, and might have some minor changes in altitude and speed, that can be due to traffic, or other external causes. Figure 4.2 shows a typical simulation flight labeled with the K-means segmentation and penalty and a typical ADS-B flight labeled with uniform segmentation and without penalty.



Figure 4.2: Results of the model with K-means segmentation and loss penalty: green areas are correctly classified, red incorrect, the faint vertical grey lines indicate the thresholds of the segments. The model performs better on the simulation data than the real data, especially for the short phases.

(a) Simulation flight

		K-means	Uniform	Uniform
		w/ penalty	w/o penalty	w/ penalty
Overall test accuracy (%)		90.86	94.16	87.29
Weighted test accuracy (%)		61.55	57.57	56.13
	TOF	6.25	12.31	-
	ICL	10.31	14.89	41.88
	CLI	97.40	96.29	95.82
Precision	CRZ	96.55	98.30	96.91
(%)	DST	98.36	99.49	99.70
	APR	13.55	21.59	6.94
	LDG	0.18	0	0
	Average	46.09	48.98	56.88
Recall (%)	TOF	100	100	0
	ICL	33.33	34.57	82.72
	CLI	88.68	95.04	93.09
	CRZ	97.82	98.22	98.11
	DST	86.03	91.01	71.74
	APR	33.14	44.29	42.00
	LDG	100	0.00	0.00
	Average	77.00	66.16	55.38

Table 4.5: The models' performance on ADS-B test data. Compared to the simulation data (table 4.3) there is a decrease of 6% overall test accuracy and 33% weighted test accuracy on average for each model.

4.2.1 Comparison with fuzzy logic model

Although the performance of the model for ADS-B data is not as high as for the simulation data, the inaccuracy mainly lies in the short phases, which are not considered in the comparison between this work and the fuzzy logic model. This subsection compares the best models, for performance on the ADS-B data, in this work with the model published by Sun et al. [41]. The model by Sun et al. is publicly available², as such, the source code was used to run this model on the ADS-B flights with available labels considered in this work. As can be seen in table 2.1b in chapter 2, different flight phases have been used for the two approaches. A comparison is only possible for the phases they have in common: ground, climb, cruise, and descent. For this purpose, the interpretation of the flight phases is slightly modified:

- The initial climb and approach phases used in this work are considered part of climb and descent.
- The portion of the flight labeled as take-off and landing is excluded from the accuracy calculation (and indicated as NA in the figures).
- The level phase used by Sun et al. is considered, respectively, climb or descent, based on the phases surrounding it.

The fuzzy logic phase identifier includes simple denoising with Density-based Spatial Clustering of Applications with Noise (DBSCAN) and is not designed to work with missing values, as such the flights analysed with it are interpolated with linear interpolation and cut at the edges where data is missing before feeding them to the tool. These results obtained by applying the preprocessing and segmentation used in this work are also reported. The number of clusters for the segmentation in this case is 59 in order for the average cluster length to be the same as the original 60 seconds used by the authors. The preprocessing and K-means segmentation do not generally have a positive impact on the fuzzy logic model, however, they do have a positive impact on the average phase precision. This is because K-means segmentation is designed to allow for more precise identification of the boundaries between the phases, more specifically when keeping a uniform segmentation of 60 seconds the average distance to the phase boundary is 15 seconds, with an average phase length of 1381 seconds this accounts for 1.1% of the error. This effect vanishes in the model of this work since the average length of a segment is 22 seconds, which means that in the uniform model the segmentation accounts on average for 0.4% of the total error. From this follows the hypothesis that the penalty and segmentation contribute to a more accurate identification of the shorter phases as they are more subject to error due to an overlap of two phases in one segment. The noise introduced by the segment length feature, however, inhibits this.

As can be seen in table 3.5 the take-off and landing phases are the ones that mostly rely on smaller segments and they are not taken into account in this comparison. While the recall of these two phases is high in the K-means segmentation

²https://github.com/junzis/flight-data-processor accessed on: 15/03/2021

Model	Accuracy (%)		Precision	Recall	
Model	Overall	Weighted	(%)	(%)	
Fuzzy logic	95.02	96.34	97.50	95.17	
Fuzzy logic	04 50	96 38	08.23	94.52	
preprocessing & K-means segmentation	54.00	50.50	30.20		
LSTM (w/penalty)	94.78	95.87	97 54	94.20	
K-means segmentation		50.01	51.04		
LSTM (w/ penalty)	86.8	80.75	90.06	89.44	
Uniform segmentation	00.0	05.10	50.00	03.44	
LSTM (w/o penalty)	07.18	97.67	08 21	97.12	
Uniform segmentation	51.10	31.01	30.21		

Table 4.6: Comparison of model in this work to the fuzzy logic approach by Sun et al. [41], with the preprocessing and segmentation in this work and without. The precision and recall refer to the average over the phases considered for the comparison (taxi, climb, cruise and descent). The model with uniform segmentation and no penalty performs best in overall accuracy and weighted accuracy.

model their precision is very low. This indicates that the false positives are more than the false negatives. With uniform segmentation on average 45 seconds were wrongly identified as take-off or landing, whereas with K-means segmentation these were 96 seconds, this introduces a 1.21% error in the overall accuracy. These results are the exact opposite to what was observed for the simulation data, pointing more towards the noise introduced by the sequence length.

Upon analysing the correlation between the quality of the data and the prediction accuracy, it is worth mentioning that there is no correlation between the overall accuracy and the quality of the data. As is stated in chapter 3, when preprocessing the ADS-B data a quality statement was made for each flight that indicates how many values of the raw transmission data were not valid data points. The data preprocessing is designed on unlabeled data and thus independent of the performance of the classification, the quality of the preprocessing compared to the DBSCAN's integrated noise elimination, used by Sun et al., shows visual improvements, which are not reflected in the classification. Figure 4.3 shows an example of the same flight, originally having bad quality, labeled with the fuzzy logic model and with uniform segmentation and without penalty model. This figure shows confirmation that, as has already been observed in table 4.6, the model developed by Sun et al. is not affected by the noise of the data.



(a) Fuzzy logic model

Figure 4.3: Classification performance on a noisy flight: comparison between fuzzy logic [41] and this work (uniform segmentation, without penalty). The difference in quality of the preprocessing can be observed in the descent phase but does not impact the accuracy of the phase identification.

Chapter 5 Conclusion

Supervised learning improves the identification of flight phases on trajectory data by 2.16% compared to the state-of-the-art model [41], as can be seen in table 4.6.

This work describes a model for classifying sequential data where the exact time steps of the data are not important but rather the succession of events in a sample. The design consists of an Long Short-Term Memory (LSTM) that takes a full sample divided into non-uniform length segments based on the K-Means principle, the labels for all segments are computed and transferred back to the single data points of the sample. In this case, the sample is a flight and the classes are flight phases (taxi, take-off, initial climb, climb, cruise, descent, approach, and landing). As flight phases have variable length, the class imbalance problem is tackled with a penalty function that adds to the loss function and consists of the average of the false negative rate and false discovery rate of all phases.

5.1 Benefits of the model

The different variations of the model described in this work bring several benefits to flight phase identification on trajectory data. The variant of the model that uses uniform segmentation and no penalty obtains a 2.16% improvement on the overall accuracy of a flight, on average, compared to the state-of-the-art method based on fuzzy logic, developed by Sun et al. This comparison requires the flight phases to be limited to 4 out of 5 main phases considered so far, for this task (see table 2.1b in chapter 2): taxi, climb, cruise, and descent. This work, however, allows for the identification of 8 different phases, most notably including the takeoff (TOF) and landing (LDG) phase, that so far have not been provided by flight phase identification models for trajectory data, but that are essential for many applications including accident avoidance [1] and analysis of noise emissions [50]. Furthermore, the flight phases used in this work adhere to the International Civil Aviation Organisation (ICAO) standard which allows this work to be used in combination with other sources that use flight phases, such as accident databases or maintenance reports.

There are slight variations of the performance of the model depending on what

performance factor is prioritised, based on the usage of the flight phases. As is frequent in imbalanced classification tasks, there is a trade-off between the overall performance and the performance of each phase [20].

Some applications will require the identification of a full flight to be able to study the behaviour of the aircraft itself, or even the pilot flying it, throughout the full flight. Each flight phase is characterised by different usage of the aircraft's actuators, which leads to a different wear of these, as such the Deutsches Zentrum für Luft- und Raumfahrt (DLR) intends to use the identification model for predictive maintenance. Scenarios like these prioritise the overall test accuracy in which case 97.19% is achieved by the model that uses the K-means segmentation and adds no penalty to the loss function.

Other aviation applications might use this model to extract sections of the flight of interest for specific applications. Such applications include but are not limited to: noise emission analysis [50], fuel emission analysis [15] and analysis of actuators that are only used in certain phases¹ such as the landing gear. To extract one or more phases, a higher accuracy for each phase is preferred. As such, a model that was trained with the loss penalty function should be used, which increases the sum of average precision and recall by more than 3%. The enforcement of the succession of flight phases was introduced for this purpose: to allow for further sectioning of the flight according to the phases. When taking the maximum output for each segment, one has to take into account that for the model the phases are rather probability-like values given by the logarithm of the Softmax, this means that there might be data points that within one phase that have just a slightly higher probability towards one phase than the points around them. By not adjusting for such phenomenon, the segmentation would be interrupted, which is not desired.

5.2 Limitations of the model

As the number of real trajectory flights with corresponding ground truth was limited to 7, the model was trained on simulation data, for which the results are positive for all flight phases. As can be seen in tables 4.3 and 4.5 in chapter 4 the results greatly vary between the simulation data and Automatic Dependent Surveillance Broadcast (ADS-B) data. The simulation results show that there is a potential for including flight phases that are so far not considered by flight phase identification models that use trajectory data. These flight phases, however, are more sensitive to the specific and more detailed features of the data that need to be trained.

As machine learning models often encounter difficulties in generalisation or insufficient training data, the hypothesis is that the problem does not lie in the ADS-B data per se but rather in the difference between the simulation and the real data. The quality of the ADS-B data does not correlate with the performance of the classification, however one can notice clear differences between the real data and the simulation data:

¹prospect usage of the model by the DLR

- The flights have different lengths, the average real flight is 59 minutes, whereas the average simulation flight is 29 minutes long.
- The transition between climb and cruising phase has a clear edge in the altitude of the real data, however, is very gradual in the simulation data.
- The descent phase takes up a large portion of the flight and consists of multiple steps in the simulation data, however, in the real data it is mostly performed with no or only minor interruptions.
- The changes both in altitude and speed span over a longer time and are generally more smooth in the simulation data than in the real data.

Although the effects of such differences can not be established with certainty, table 4.5 shows how the difference in performance mainly lies in the shorter phases and is reduced using uniform segmentation rather than k-means segmentation. From this observation, it is hypothesized that the relatively small changes in the real data, originating from events such as air traffic, have the effect of a big change when given to a model trained on simulation data. This could lead to the network expecting a change in phase when there actually is none. Furthermore, the segment length and relative conceptualisation of time have a negative impact on the performance as the simulation data appears to be not representative of the real flight times, which changes the balance of the average number of segments belonging to a flight phase. This effect is further amplified when training the model to use the time of each segment as a feature of that segment, the differences in performance with and without this feature are also observed in the simulation performance. As such the uniform segmentation, that omits this feature performs better than the K-means segmentation, where this feature is necessary to keep the scale of each segment.

Another limitation is that the flight phases are defined in their details by the DLR, although they adhere to the ICAO standard, the specific rule set is not defined or standardised in the field of aviation. Furthermore, the flight phases are defined for short flights as the model is designed for European flights. This implies that future applications of this model by other institutes that use different parameter values need to redefine the flight phases and train the model with slightly different labels or data. However, it is hypothesised that slight changes in the definitions will obtain similar performance. There are also further flight phases and sub-phases that are not used in this work, such as the standing phase, rejected take-off, change of cruise level, maneuvering, and the different sub-phases of the approach and landing. These phases were not used mostly because they do not apply to commercial aircrafts but also for simplification or lack of data due to their infrequency.

5.3 Future Work

To further improve the performance of the variants of the model, future work should address two main points: the lack of real flights and improvement of the features.

5.3.1 Supplementary Real Flight Data

The main limits of the model lie in the data it is trained with, the availability of more real flights could allow transfer learning. Transfer learning means that the model is trained initially on simulation data but at a later stage transferred to real data and further trained, as has been proven successful for other applications [45, 24].

With the sensitivity and lack of availability of these real flights it should be taken into consideration to create a generative deep learning model that is able to either generate data [47] or transform the simulation data into data that is more similar to the ADS-B data. The generation of data could consist in creating labeled flights given two airports, while the transformation of simulation data could be performed by using the introduction of noise and applying changes to its trajectory curves. Given the lack of improvement of accuracy of the fuzzy logic tool when used in combination with more elaborate preprocessing and K-means segmentation indicates that future work should consider different preprocessing approaches. If the model is trained on ADS-B (or real-like) data, further improvement could be achieved by integrating the preprocessing into the model for it to be integrated in the training process.

5.3.2 Feature Improvement

Aside from the use of ADS-B data for training, future work should also consider using the rate of climb variable as a feature rather than computing it from the altitude. This work computed the altitude as was done for the labeling tool provided by DLR. However, this approach might neglect features that could contribute to a better identification: the rate of climb is a significant factor for the flight phases, as such it is used in other works that identify them [41, 26]. The differences between the computed rate of climb and the actual rate of climb variable can be seen in figure 5.1. This figure shows that the computation of the rate of climb does not capture the dynamics over time, which could be used both for classification but also for the K-means segmentation. In fact, the computed rate of climb was not used for the K-means segmentation as it is rather static. Exploration of the use of the computed rate of climb found that it would not contribute positively to the segmentation when considering the maximum possible accuracy. This is likely not the case for the original rate of climb variable. It is hypothesised that by including a global concept of time, by normalising over the whole dataset which carries the downside of having to determine a maximum flight length, and the rate of climb



Figure 5.1: Comparison between computed rate of climb and original rate of climb variable. Features of the actual rate of climb variable are lost when using the computation from the altitude variable.

variable, the models that use K-means segmentation will show an improvement in performance.

Chapter 3 describes how the features of the segments are normalised for each flight, rather than the full dataset. This is done to avoid the data real input values to be out of the range of the network, as the simulation data is not in all factors representative of the real data. As mentioned in the previous section (5.2) the main feature impacted by this is the segment length. Future work shall explore the use of parameters derived from physics to normalise the variables. These physical parameters are the maximum possible altitude, maximum length of a flight within Europe, and maximum speed.

5.4 Applications in Other Fields of Research

The model presented in this work is described and analysed in the field of flight phase identification for aviation, however, it can be used for classifying different sources of sequential data. This work aims at the classification of sequential data where the exact time steps of the data are not important, but rather the succession of events in a sample is considered. The main advantage is that it can focus the attention on events of different lengths by dividing the sequence into non-uniform segments, based on the amount of change present. The segmentation itself allows for the application of the model in other sequence classification tasks that suffer from class imbalance. The class imbalance is further addressed by adding a penalty to the loss function, computed from the average precision and recall of all phases.

To transfer this model to other fields of research, examples can be drawn from a robot learning from experience or observation, such as the Neuro-Inspired Companion (NICO)[22].

The movements of a robot are a different type of trajectories, characterised by the different actuator values of the robot over time. These trajectories are also divided into phases, for the example of grasping an object: visually identifying the object, reaching out for it, grasping it, and retrieving it. While often these phases are predefined in the source code of the robot, modern research investigates the possibilities of robots learning from humans [21]. This could either be done through the robot observing the human but also through the human guiding the robot's movements. Both cases would benefit from the robot being able to identify the phases of the grasping movement it is learning. This could help to, for example, regulate the speed of its motion, but also to learn what actuators and sensors are of importance to which phase and how much they are utilised. Similar to what is done in aviation, this knowledge could also be used in post analysis to analyse the usage of the actuators and perform predictive analysis to prevent actuator failure during the crucial moments of a Human-Robot Interaction (HRI) experiment.

Classifying objects based on the observation of dropping them on a surface is another example of such applications. In this case, the robot receives both the visual and auditory input of the falling object. It might be of use to segment these signals into different phases such as: holding the object, dropping the object, first contact of the object with the surface, and potential bouncing of the object. Such phases, similar to flight phases, will be of different length and their transition can be identified by a great amount of change in the variables (such as motor noise or movement of the object). By extracting information on each of these phases, the robot could learn to identify general features of objects such as a "heavy hard bouncy object" or a "light soft rolling object". By first learning to identify each phase and extract features from them the robot could learn to further generalise its knowledge to objects it has not yet seen. A downside to using this model is that it requires supervised data for the classification of phases.

Another task suited for a model like the one presented, is a gesture recognition task. Gestures consist of a succession of events over time, however, not all events nor complete gestures have the same length. When considering the gestures described by Jirak et al. [18], the authors state that one of the difficulties in a task is the variable length of the gestures. Considering a continuous signal of gestures, the pauses in between gestures are long segments with low variability in the signals, however, the gestures themselves and the pauses within a gesture will be shorter as they are characterised by high variability in the signals. By analysing segments of non-uniform length, the classification can be optimised by providing more computation (i.e. shorter segments) in the moments of high variability, reducing the total sequence length. This task is suitable for a LSTM as the model with highest accuracy described by the authors is a LSTM. It is worth exploring the benefits of combining the K-means segmentation with the existing LSTM presented by the authors.

This work has seen positive results for flight phase identification, possibilities for further improvement, and usability in many applications, in aviation and other domains. In summary, it provides a solid basis for future research.

Acronyms

- **ADREP** Accident Data Reporting. IX, 2, 19, 20
- ADS-B Automatic Dependent Surveillance Broadcast. IV, IX, 1, 3, 4, 15, 16, 18–20, 22, 25, 29, 30, 34–40, 44
- **APR** approach. 20–22, 24, 30, 35, 36, 38
- CLI climb. 3, 20–22, 24, 30, 35, 36, 38
- **CRZ** cruise. 20–22, 24, 30, 35, 36, 38
- **DBSCAN** Density-based Spatial Clustering of Applications with Noise. 5, 6, 11, 17, 39, 40
- DLR Deutsches Zentrum für Luft- und Raumfahrt. 17, 18, 21, 26, 44–46
- **DST** descent. 20–22, 24, 30, 35, 36, 38
- **FAA** Federal Aviation Administration of the United States Department of Transportation. 19
- GMM Gaussian Mixture Model. 10, 13
- HRI Human-Robot Interaction. 48
- **IATA** International Air Transport Association. 2
- ICAO International Civil Aviation Organisation. IV, IX, 2, 3, 19–21, 43, 45
- ICL initial climb. 20–22, 24, 30, 35, 36, 38
- LDG landing. 20–22, 24, 30, 35, 36, 38, 43
- LSTM Long Short-Term Memory. IV, V, VII, 1, 4, 6–8, 10, 13, 23, 26, 43, 48
- NICO Neuro-Inspired Companion. 47
- **NN** Artificial Neural Network. VII, 6, 7, 26

NODE Neural Ordinary Differential Equations. 10, 13

RNN Recurrent Neural Network. VII, 6, 7

SVM Support Vector Machine. 10, 13

TOF take-off. 3, 20–22, 24, 30, 35, 36, 38, 43

TXI taxi. 20–22, 24, 30, 35, 36

Bibliography

- [1] Boeing Commercial Airplanes. Statistical summary of commercial jet airplane accidents. volume 2019, 1959.
- [2] R. Alligier, D. Gianazza, and N. Durand. Learning the aircraft mass and thrust to improve the ground-based trajectory prediction of climbing flights. volume 36, pages 45–60, 2013.
- [3] H. Aydın, Ö. Turan, T. Hikmet Karakoç, and A. Midilli. Exergo-sustainability indicators of a turboprop aircraft for the phases of a flight. volume 58, pages 550–560, 2013.
- [4] T. Baklacioglu. Modeling the fuel flow-rate of transport aircraft during flight phases using genetic algorithm-optimized neural networks. volume 49, pages 52–62. Elsevier, 2016.
- [5] H.B. Barlow. Unsupervised learning. volume 1, pages 295–311, 1989.
- [6] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. volume 57, pages 289–300. Wiley Online Library, 1995.
- [7] A. Buccini, O. De la Cruz Cabrera, M. Donatelli, A. Martinelli, and L. Reichel. Large-scale regression with non-convex loss and penalty. volume 157, pages 590–601, 2020.
- [8] F. Caliva, C. Iriondo, A. Morales Martinez, S. Majumdar, and V. Pedoia. Distance map loss penalty term for semantic segmentation. 2019.
- [9] M.J. Carrico. Operational flight phase determination and indication system. Google Patents, 2012. US Patent 8,217,807.
- [10] Y.S. Chati and H. Balakrishnan. Analysis of aircraft fuel burn and emissions in the landing and take off cycle using operational data. In *International Conference on Research in Air Transportation*, 2014.
- [11] N. Chawla, N. Japkowicz, and A. Kołcz. Special issue on learning from imbalanced data sets. volume 6, pages 1–6, 2004.

- [12] H.-J. Chin, A. Payan, C. Johnson, and D.M. Mavris. Phases of Flight Identification for Rotorcraft Operations. AIAA Scitech 2019 Forum, 2019.
- [13] M. Crossley. A guide to coordinate systems in great britain. 1999.
- [14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.
- [15] A. Filippone, B. Parkes, N. Bojdo, and T. Kelly. Prediction of aircraft engine emissions using ads-b flight data. pages 1–25. Cambridge University Press, 2021.
- [16] Mohamad H Hassoun et al. Fundamentals of artificial neural networks. MIT press, 1995.
- [17] S. Hochreiter and J. Schmidhuber. Long short-term memory. volume 9, pages 1735–1780, 1997.
- [18] D. Jirak, S. Tietz, H. Ali, and S. Wermter. Echo state networks and long short-term memory for continuous gesture recognition: a comparative study. pages 1–13. Springer, 2020.
- [19] C. Johnson, A. Rusu, A. Breitzman, J. Bucknam, and N. Laposta. Phase of flight and rule of flight calculator. In 7th International Conference on Research in Air Transportation, 2016.
- [20] B. Juba and H.S. Le. Precision-recall versus accuracy and the role of large data sets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4039–4048, 2019.
- [21] M. Kerzel, T. Pekarek-Rosin, E. Strahl, S. Heinrich, and S. Wermter. Teaching nico how to grasp: An empirical study on crossmodal social interaction as a key factor for robots learning from humans. volume 14, page 28. Frontiers, 2020.
- [22] M. Kerzel, E. Strahl, S. Magg, N. Navarro-Guerrero, S. Heinrich, and S. Wermter. Nico—neuro-inspired companion: A developmental humanoid robot platform for multimodal interaction. In 2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), pages 113–120. IEEE, 2017.
- [23] S. Kovarik, L. Doherty, K. Korah, B. Mulligan, G. Rasool, Y. Mehta, P. Bhavsar, and M. Paglione. Comparative analysis of machine learning and statistical methods for aircraft phase of flight prediction. 2020.
- [24] B. Kustowski, J. A. Gaffney, B. K. Spears, G. J. Anderson, J. J. Thiagarajan, and R. Anirudh. Transfer learning as a tool for reducing simulation bias: Application to inertial confinement fusion. volume 48, pages 46–53, 2020.

- [25] S.Z. Li and A. Jain. False Negative Rate, pages 407–407. Springer US, Boston, MA, 2009.
- [26] D. Liu, N. Xiao, Y. Zhang, and X. Peng. Unsupervised flight phase recognition with flight data clustering based on gmm. In 2020 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), pages 1–6, 2020.
- [27] R. Longadge and S. Dongre. Class imbalance problem in data mining review. volume abs/1305.1707, 2013.
- [28] J. MacQueen. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, pages 281–297, Berkeley, Calif., 1967. University of California Press.
- [29] D. Michie, D.J. Spiegelhalter, C.C. Taylor, et al. Machine learning. volume 13, pages 1–298. Technometrics, 1994.
- [30] J. Naganawa and H. Miyazaki. A method for accurate ads-b signal strength measurement under co-channel interference. In 2018 Asia-Pacific Microwave Conference (APMC), pages 354–356, 2018.
- [31] D.L. Olson and D. Delen. Advanced data mining techniques. Springer Science & Business Media, 2008.
- [32] M. Paglione and R. Oaks. Determination of Horizontal and Vertical Phase of Flight in Recorded Air Traffic Data.
- [33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. De-Vito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 32, pages 8024–8035. Curran Associates, Inc., 2019.
- [34] M. Poppe and J. Buxbaum. Clustering climb profiles for vertical trajectory analysis. In *Proceedings of the 10th SESAR Innovation Days*, 2020.
- [35] L. Purton, H. Abbass, and S. Alam. Identification of ads-b system vulnerabilities and threats. 2010.
- [36] D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning representations by back-propagating errors. volume 323, pages 533–536. Nature Publishing Group, 1986.

- [37] A. Scharnweber and S. Schier. A unified flight phase and event model for the analysis of heterogeneous atm simulations. volume 2, pages 171–183. Springer, 2011.
- [38] M. Schäfer, M. Strohmeier, V. Lenders, I. Martinovic, and M. Wilhelm. Bringing up opensky: A large-scale ads-b sensor network for research. In *Proceed*ings of the 13th International Symposium on Information Processing in Sensor Networks (IPSN '14), pages 83–94, Piscataway, NJ, USA, April 2014. IEEE Press.
- [39] Z. Shi, M. Xu, Q. Pan, B. Yan, and H. Zhang. Lstm-based flight trajectory prediction. In 2018 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2018.
- [40] M. Strohmeier. Research usage and social impact of crowdsourced air traffic data. volume 59, 2020.
- [41] 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.
- [42] J. Sun, J. Ellerbroek, and J. Hoekstra. Modeling and inferring aircraft takeoff mass from runway ads-b data. In 7th International Conference on Research in Air Transportation, 2016.
- [43] J. Sun, J. Ellerbroek, and J. Hoekstra. Aircraft initial mass estimation using bayesian inference method. volume 90, pages 59–73. Elsevier, 2018.
- [44] J.M. ten Have. The development of the nlr atc research simulator (narsim): Design philosophy and potential for atm research. volume 1, pages 31 – 39, 1993.
- [45] H. Tercan, A. Guajardo, J. Heinisch, T. Thiele, C. Hopmann, and T. Meisen. Transfer-learning: Bridging the gap between real and simulation data for machine learning in injection molding. volume 72, pages 185–190, 2018. 51st CIRP Conference on Manufacturing Systems.
- [46] C. Uri. Notes on the flight phase taxonomy. 2018.
- [47] J. Xu, H. Li, and S. Zhou. An overview of deep generative models. volume 32, pages 131–139. Taylor & Francis, 2015.
- [48] L.A. Zadeh. Fuzzy sets. volume 8, pages 338–353, 1965.
- [49] T. Zeh, J. Rosenow, R. Alligier, and H. Fricke. Prediction of the propagation of trajectory uncertainty for climbing aircraft. In 2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC), pages 1–9. IEEE, 2020.

- [50] C. Zellmann, B. Schäffer, J. M. Wunderli, U. Isermann, and C. O. Paschereit. Aircraft noise emission model accounting for aircraft flight parameters. volume 55, pages 682–695, 2018.
- [51] Y. Zhu, X. Shen, and W. Pan. Network-based support vector machine for classification of microarray samples. volume 10, pages 1–11. BioMed Central, 2009.

Erklärung der Urheberschaft

Hiermit versichere ich an Eides statt, dass ich die vorliegende Masterthesis im Studiengang Intelligent Adaptive Systems selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel - insbesondere keine im Quellenverzeichnis nicht benannten Internet-Quellen – benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus Veröffentlichungen entnommen wurden, sind als solche kenntlich gemacht. Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der auf dem elektronischen Speichermedium entspricht.

Ort, Datum

Unterschrift

Erklärung zur Veröffentlichung

Ich stimme der Einstellung der Masterthesis in die Bibliothek des Fachbereichs Informatik zu.

Ort, Datum

Unterschrift