Predicting Deviation of Flight Entry into Air Sector using Machine Learning Techniques

Christian Klötergens
University of Hildesheim
Hildesheim, Germany
kloetergens@uni-hildesheim.de

Cristina Acevedo
University of Hildesheim
Hildesheim, Germany
acevedo@uni-hildesheim.de

Indra Firmansyah
University of Hildesheim
Hildesheim, Germany
firmansyah@uni-hildesheim.de

Leonardo Antiqui
University of Hildesheim
Hildesheim, Germany
antiqui@uni-hildesheim.de

Kiran Madhusudhanan
University of Hildesheim
Hildesheim, Germany
kiranmadhusud@ismll.de

Mohsan Jameel
Deutsches Zentrum für Luft- und Raumfahrt
Braunschweig, Germany
mohsan.jameel@dlr.de

***Abstract*—The management of air traffic is a complex task that requires ensuring the safety and efficiency of aircraft trajectories when transiting from one airspace sector into another. This work explores the use of historical flight data to predict if a flight will commit to the planned entry point when entering an airspace sector. To achieve this, we propose a feature engineering method that can be employed to convert raw flight data into a matrix which captures flight count information in predefined grids. This matrix is referred to as the Air Space Occupancy Grid (ASOG) and it captures the state of traffic in an airspace sector and its immediate vicinity. Experiments are performed using the Swedish Civil Air Traffic Control (SCAT) dataset. To predict whether an aircraft will deviate from its planned entry point, supervised machine learning algorithms are used to train a model. Through experiments on real-world data, we showcase that ASOG provides a systematic way of incorporating the state of the airspace sector and improving the performance of prediction models compared to simple features. The prediction output can be used to notify human air traffic controllers in advance about potential deviation to flight plan upon entry to an airspace sector. This can improve the planning process of air traffic controllers in their work in maintaining safe and efficient air traffic.**

I. Introduction

The primary objective of air traffic control (ATC) is to regulate the flow of air traffic in order to maintain a safe and efficient air traffic flow [1]. The main responsibility of air traffic controller officers (ATCO) is to avoid dangerous situations within their assigned airspace sectors. Factors such as flight rerouting and weather conditions introduce uncertainty to the air traffic load in an airspace sector. This creates high demand for efficient control and could put ACTOs in a setting with higher stress compared to the planned scenario [2]. Furthermore, maintaining safety in such environments is crucial, therefore ATCOs have increasingly high responsibilities which must be carried out assuring the highest performance. For instance, a flight handover between airspace sectors can become a complex task if the flight deviates from the agreed entry point, thereby triggering a sequence of tasks that include verifying conflicts for the new route. Such occurrences have implications for the flow of traffic within the sector and the negotiation between sectors. The ability to anticipate deviations from flight plan is important, as alterations to one flight's trajectory can create widespread repercussions on numerous other flights, particularly when traffic is high [3]. As the volume and intricacy of air traffic escalate, the frequent coordination required for handovers leads to an increased workload for ATCOs.

The advances in Artificial Intelligence and Machine Learning techniques have opened avenues to develop automated assistance systems for ATCOs, supporting them in different routine tasks [4]–[6]. In this paper, we propose a prediction task in the area of ATC, in which machine learning techniques can be used to create a solution for predicting if a flight will deviate from the latest flight plan upon entry to an airspace sector. The prediction is formulated as a binary classification problem, namely, for a given flight instance, it outputs one of two classes: deviation or no deviation. Our prediction model takes as input some features of the state of the flight of the entering aircraft, such as its current location, and its planned trajectory. However, these inputs only capture the dynamics of the problem partially. Therefore, we developed a framework to incorporate the state of traffic within an airspace sector and its surroundings. We propose the Air Space Occupancy Grid (ASOG), a group of grids mapping the airspace sector. ASOG is used to keep track of how many aircraft are flying in certain areas at different times. We hypothesize that using ASOG improves the performance of Machine Learning models that predict deviation in the flight entry point to an airspace sector.

In the experiments, we evaluate the use of ASOG on the performance of the binary classification model trained for the proposed task. We demonstrate that the model that achieves the best performance incorporates ASOG during the model training process.

Since air traffic and ATCOs' workload are expected to grow significantly in the near future [6], the task proposed in this paper is relevant to support real time decision making of ATCOs. Entry point deviation prediction allows the likelihood of entry point deviation to be included in the planning process of ATCOs in maintaining safety within the airspace sector.

The main contributions of this work are:

- We formulate a prediction task relevant to ATC. The task is to forecast whether a flight will deviate from the latest planned trajectory when entering an airspace sector. Machine Learning methods are used to train a model for this prediction task.
- We propose a framework to construct a feature called Air Space Occupancy Grid (ASOG) from flight data to capture the state of the air traffic in the vicinity of an airspace sector. We demonstrate that this feature improves the performance of models built for the prediction task previously mentioned.
- We perform experiments using the Swedish Civil Air Traffic Control (SCAT) dataset to compare the performance of models trained with and without ASOG.

II. Related Work

Research in AI applied to air traffic management (ATM) and ATC has been carried out for decades. R.B. Wesson was the first to apply AI techniques to ATC in the 70s [7]. Approaches to develop conflict detection and resolution automation tools have evolved from using mathematical algorithms, modeling and optimization approaches using heuristics and constraint programming, to new perspectives using machine learning and deep reinforcement learning (DRL) [8]. AI research in ATM is a growing domain. The number of publications of AI in ATM has almost doubled between 2014 and 2018, and has more than tripled since 2010 [9]. Most publications targeting the ATCO focused on predicting or analyzing their behavior and their decisions. A brief literature review shows that AI prediction in ATM is performed using a vast range of machine learning models, with the most utilized being: Multi-Agent Systems [10], [11], Neural Network [12], [13], Random Forest [14], [15], Gradient Boosting Machine [13], [15], and Support Vector Machine [15]. Although the adoption of AI in ATM has been relatively slow compared to other industries, such as finance and healthcare, due to a number of factors such as safety and regulatory concerns, lack of data sharing, and the complexity of the ATM system, there is increasing interest and investment in this area [9].

Previous machine learning for ATC have addressed prediction in different ways. Pham et al. [14] introduced a paper proposing a method for extracting ATCO speed, direction and altitude commands from flight data and train classification and regression prediction models. Such commands were extracted from the trajectory data and the classification target was generated by thresholding the ATCO commands. Ma and Tian [16] created a model using neural networks for a different task in order to predict 4D aircraft trajectory. On this approach trajectories were preprocessed in order to obtain sufficient data and then a model was trained in order to predict step by step trajectories which reflected the actual flight data. Chen et al. [17] have approached another prediction task by forecasting traffic flow using neural networks and ensemble models. Here, traffic flow data is used to predict the future flow at different times. These models have been able to implement modern tools to provide insights that could improve air travel and ATCO tasks.

Occupancy grid maps (OGMs) are a type of spatial data structure that represents the environment as a 2D grid of cells. Each cell in the grid represents a small area of the environment and can be assigned a binary value indicating whether it is occupied or not by an object of interest. Occupancy grids were first proposed by H. Moravec and A. Elfes in 1985 [18]. They are commonly adopted for probabilistic localization and mapping in robotics and autonomous systems. There have been some attempts to use OGMs for path planning and conflict detection in the air traffic control field. For example, Jardin [19] presented an algorithm for strategic conflict detection for ATC based on the use of a 4-dimensional space and time grid to represent the airspace, similar to the one proposed in this paper. Similarly, Ayhan and Samet [20] describe a novel stochastic trajectory prediction approach for ATM that considers the airspace as a 3D grid network of cubes. Alam et al. [21] discretize the airspace in equal sized hyper rectangular cells that act as a repository of the airspace information such as weather, atmospheric properties, intent information of aircraft, etc. Hong Liu et al. [22] proposed Traffic Situation Graphic (TSG) generated by splitting the 3D earth space with fixed grid map and flight levels. Baspınar et al. [23] presents an optimization-based autonomous ATC system based on integer linear programming (ILP) constructed via a mapping process that contains discretization of the airspace with predicted trajectories to improve the time performance of conflict detection and resolution.

III. Problem Formulation

A. Problem and data available

An aircraft heading to its destination crosses from one en-route airspace sector to another, following a route according to its predefined flight plan. When the aircraft approaches the borders of the incoming airspace sector, the controller of the outgoing airspace sector and the controller of the incoming airspace sector execute a hand-off process. Then, the planning controller of the incoming sector uses the latest flight-plan information and other variables of the aircraft to detect future potential conflicts in their sector, either with other aircraft or other obstacles like weather, restricted airspace, etc. The look-ahead time window of the planning controller is normally between 10 to 20 minutes. When a potential conflict is detected, the controller needs to decide which aircraft must be rerouted. One alternative is to command the aircraft to change the entry point to the sector. This difference with respect to the original entry point is called a flight deviation in this paper. We propose machine learning models capable of predicting if an aircraft will enter a sector at the planned entry point or if, on the contrary, it will be deviated. The nature of this problem is binary and we propose binary classification machine learning models trained with historical ADS-B data to solve it.

The ADS-B system provides accurate and real-time aircraft position information. 4D trajectories are an integral concept within the ADS-B system. A 4D trajectory refers to an aircraft's flight path in four dimensions: latitude, longitude, altitude, and time. These 4D trajectories in a sector and immediate surrounding areas encode decision-making patterns of the controllers of the sector and the primary goal of our machine learning models is to learn the complex pattern to support human ATCOs in decision-making process.

Although real-time aircraft position is of vital importance in the decision-making process of the ATCOs, the future state of the airspace sector is equally important to determine if a flight plan requires adjustments. The aircraft's planned trajectory provides the controllers with this window into the future. Consequently, both actual and planned trajectories constitute the training data of our machine learning models. For simplification, we do not consider the altitude in this paper. Hence, each point of actual and planned trajectories consists of latitude, longitude and time. In summary, we define these trajectories as:

- Actual trajectory: trajectory actually taken by the aircraft
- Planned trajectory: planned trajectory at certain times

Consequently, actual and planned trajectories are formally defined accordingly in (1) and (2), respectively.

$$rt\left(h_k\right) = \begin{pmatrix} \lambda_1 & \phi_1 & t_1 \\ \lambda_2 & \phi_2 & t_2 \\ \vdots & \vdots & \vdots \\ \lambda_N & \phi_N & t_N \end{pmatrix}_{N\times 3} \tag{1}$$

$$pt\left(h_k, t_s\right) = \begin{pmatrix} \lambda'_1 & \phi'_1 & t'_1 \\ \lambda'_2 & \phi'_2 & t'_2 \\ \vdots & \vdots & \vdots \\ \lambda'_M & \phi'_M & t'_M \end{pmatrix}_{M\times 3} \tag{2}$$

Here λ is latitude, ϕ is longitude and t is time. The matrix of actual trajectories, rt, is the set of N 3D points for the flight with id f_k. The matrix of planned trajectories, pt, is the set of M 3D points for the flight with id h_k at the timestamp t_s. Note that the primed coordinates correspond to planned coordinates.

B. Machine learning approach and target

Since this is a supervised machine learning setting, the training data would not be completed without the labels that reflect the final decision made by the controller. Therefore, the training data must be provided in pairs of predictors and targets. The predictors encode the situational information at the decision time and they will be elaborated in the later sections. The targets are the final decisions made by the controllers for each incoming flight. We also mentioned that these final decisions are encoded as a binary target that compares the future actual entry point of the aircraft into the sector with respect to the planned entry point at the decision time. In other words, if there was a flight deviation or not.

This deviation feature comes from the difference between two 2D positions and it is naturally a continuous variable. In this paper, this difference is measured as the geodesic distance between these two 2D points:

$$y_c\left(h_k\right) = \text{geod}\left(\left(\lambda, \phi\right), \left(\lambda', \phi'\right)\right) \tag{3}$$

Where λ and ϕ are the real latitude and longitude of the flight h_k at its entry time $h_{entry,k}$, and λ' and ϕ' are the planned latitude and longitude of the flight h_k at the forecasted entry time $h'_{entry,k}$ according to the flight plan at the prediction time $t_{s,k}$.

In this work, we decided to model the problem as a binary classification problem. This requires us to make y_c discrete. System navigation errors, wind and several other factors can cause some position variations even when the aircraft is intended to follow its flight plan. Therefore, we need to define a threshold $\delta > 0$ to classify the flights. This means, aircraft deviating at least δ from their expected entry point belong to the positive class, otherwise they belong to the negative class. This can be expressed as:

$$y_d = \begin{cases} 1, & y_c \geq \delta \\ 0, & y_c < \delta \end{cases} \tag{4}$$

To create a mathematical model of the decision-making process in air traffic control using a rule-based approach is an incredibly complex endeavor. For these cases, a data driven approach using machine learning can considerably simplify the task. This data driven approach uses the set of pairs of predictor-targets and learns from these historical data points the patterns of the decision-making process. Ultimately, the learned models are parameterized functions capable of taking new flight instances and outputting a scalar number indicating if the new instances belong to one class or the other. That is, if the incoming flights are expected to deviate (1) or not (0). Mathematically, this functions can be written as,

$$f_\theta : \mathbb{R}^M \rightarrow \{0, 1\} \tag{5}$$

where M is the dimensionality of the engineered features.

The performance of this data-learned mathematical model is measured in terms of its capability to accurately predict the target y of unseen predictors x. Here unseen means that these pairs of predictor-targets were not used as part of the training data. They are a subset of the initial available data that is commonly used to measure the accuracy and generalization of the learned models.

IV. Methodology

In this section we describe how we use flight data as presented in section III-A to solve the binary classification task as defined in section III-B using machine learning models.

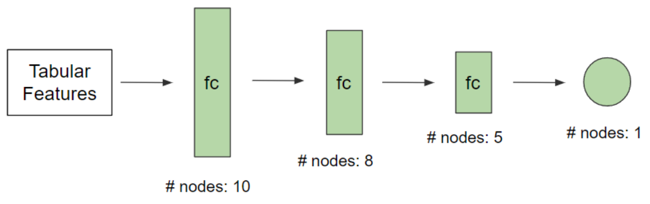


Fig. 1. Baseline NN model

A. Tabular Features

Input features are captured at prediction time, t_s. The prediction time is set with respect to the forecasted entry time. In practice, this is a time window between 10 and 20 minutes before the forecasted entry time into the sector. This difference between the forecasted entry time and the prediction time is called the buffer time b.

We define tabular features that can be inferred from the 3D trajectories by making certain assumptions. To identify forecasted entry point into an airspace sector and the corresponding forecasted entry time, it is assumed that the aircraft travel with a constant speed on a straight line between the 3D points in the predicted trajectory. We hypothesize the following features to be helpful for training the binary classification model and therefore declared them as input for our baseline models. These features are:

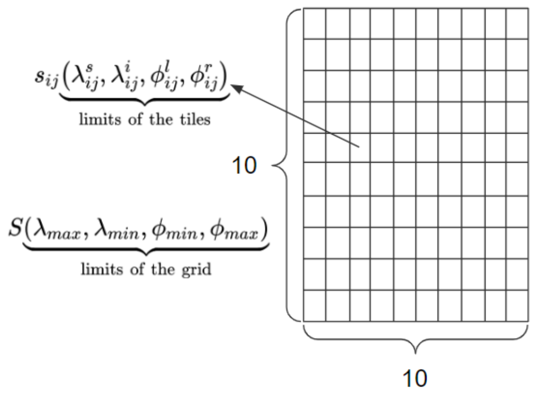
- Latitude of position at prediction time
- Longitude of position at prediction time
- Latitude of forecasted entry point into sector
- Longitude of forecasted entry point into sector
- Hour within day (0-23) from forecasted entry time
- Day within week (1-7) from forecasted entry time

B. Baseline Models

There are several types of binary classification models, including logistic regression, support vector machines, random forests, and neural networks. Each model has its strengths and weaknesses and all of them can trivially handle tabular features. Neural Networks (NN) have proven to be powerful mappings to model complex real-life problems like the one presented in this paper. eXtreme Gradient Boosting (XGBoost) [24] is known for its speed and performance and it is the go-to option in a wide variety of problems. We use these two types of models in our experiments.

Figure 1 shows the illustration of the baseline model that is based on a neural network, and it is referred to as baseline NN model. The architecture of the baseline NN model is a neural network with three hidden fully connected layers and an output layer with one node. The loss used in the output layer is binary cross entropy.

In this work, a baseline model is one that only uses tabular features for training. There are two variants of the baseline mode. One is based on a NN and another based on XGBoost. As we will see in the experimental results section, having two baseline models with different model bases allow us to evaluate the effect of including ASOG within a given model basis (NN or XGBoost).



$\underbrace{s_{ij}(\lambda^s_{ij}, \lambda^i_{ij}, \phi^l_{ij}, \phi^r_{ij})}_{\text{limits of the tiles}}$
10
$\underbrace{S(\lambda_{max}, \lambda_{min}, \phi_{min}, \phi_{max})}_{\text{limits of the grid}}$
10

Fig. 2. Grid map limits

C. Air Space Occupancy Grid

An ATCO decides to change the flight plan of an aircraft if they find a potential conflict in the current trajectory. The change of the flight plan may, in turn, change the entry point of the aircraft into an airspace sector. ATCOs consider the aircraft in the vicinity to deduce if there will be a potential conflict. This information is not captured by the tabular features and hence, this motivates us to propose ASOG, which capture the information of aircraft in the vicinity.

At prediction time t_s, the latest planned trajectory of each flight gives the planned positions of the aircraft at multiple points of time after t_s. We project this information into a 2D matrix. Hence, the 3D trajectory of each flight needs to be discretized accordingly so that the projection can be manifested. Firstly, straight lines are drawn between different positions (latitudes and longitudes) of the aircraft and each line links each aircraft position with the position of the next timestamp. The airspace sector and its immediate surrounding area is partitioned into a fixed number of equally sized rectangular cells using a 2D grid map of shape $(N \times M)$, as shown in figure 2, where N and M are set as 10. The cells were defined by their coordinate limits.

The discretization of the 3D planned trajectory of a flight in the grid defines what we call the *own occupancy matrix*. Cells that come into contact with the drawn line are assigned a value of 1, otherwise their value is 0. This can be mathematically expressed as,

$$s_{ij}\left(h_k, t_s, t'\right) = \begin{cases} 1 & \lambda^i_{ij} \leq \lambda'\left(t'\right) \leq \lambda^s_{ij}\; and\, \phi^l_{ij} \leq \phi'\left(t'\right) \leq \phi^r_{ij} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

The resulting own occupancy matrix is defined in (7).

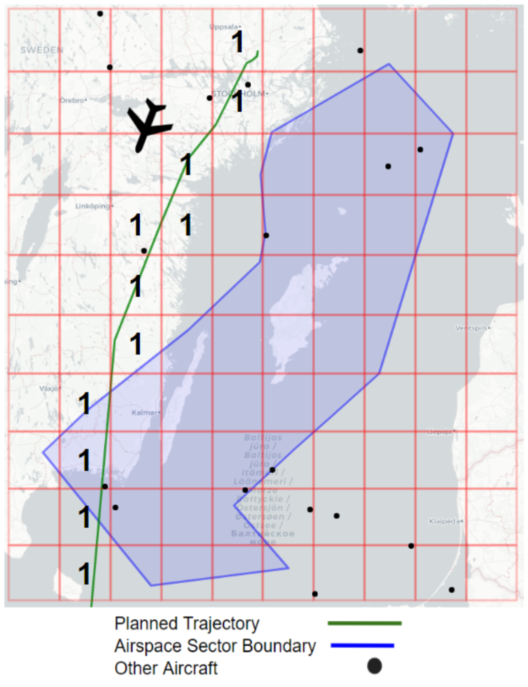


Fig. 3. Own Occupancy Matrix illustration. Any grid that is traversed by planned trajectory is assigned value of 1. Image generated is using [25]

$$S\left(h_k, t_s\right)=\begin{pmatrix} s_{1,1} & s_{1,2} & \cdots & s_{1,M} \\ s_{2,1} & s_{2,2} & \cdots & s_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ s_{N,1} & s_{N,2} & \cdots & s_{N,M} \end{pmatrix}_{N\times M} \tag{7}$$

Own occupancy matrix is illustrated in figure 3, where the grid as shown in figure 2 is superimposed on the map of Sweden. The green line signifies the planned trajectory of the aircraft. Any grid that is traversed by the green line is assigned a value of 1, which is consistent with (6).

At a given time, the positions (latitudes and longitudes) of aircraft in the vicinity of an airspace sector occupy a continuous space. Similar to *own occupancy matrix*, this information is projected into multiple 2D matrices, where each matrix corresponds to a specific point of time. Each matrix has a dimension of $N \times M$ and each cell shows the number of aircraft in that cell at a particular point of time. The first matrix captures the sector occupancy by other aircraft at the expected entry time of the flight of interest, t'_{entry}. The value of each cell is formally defined as follows:

$$s_{ij}\left(h_k, t_{s,k}, t'_{\text{entry},k}\right)=\sum_{\substack{\kappa=1\\ \kappa\neq k}}^{K} s_{ij}\left(h_\kappa, t_{s,k}, t'_{\text{entry},k}\right) \tag{8}$$

There are V time periods in which the occupancy of other aircraft in the vicinity is captured. This results in a tensor that we referred to as *video sequence*. Figure 4 illustrates one matrix in the video sequence. The value assigned to each cell

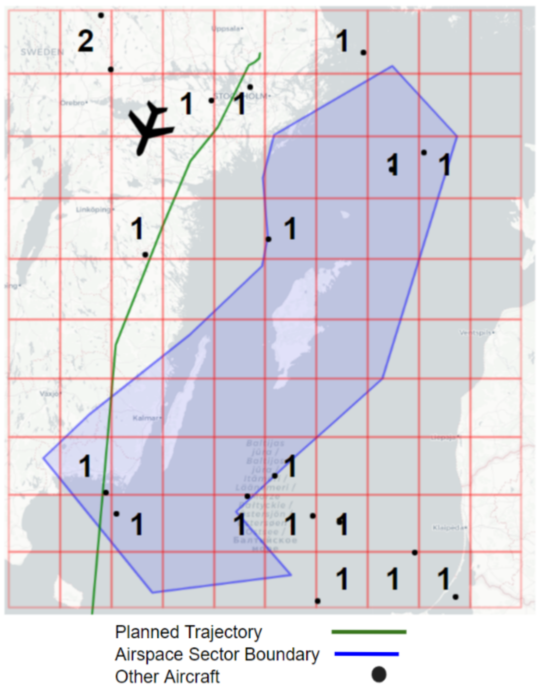


Fig. 4. Illustration of one matrix in Video Sequence. One matrix corresponds to one time period captured in Video Sequence. If there is at least one aircraft in a grid, the count of the aircraft in the grid is shown. Image is generated using [25]

corresponds to the number of aircraft in that cell, which is consistent with (8). Video sequence is formally defined as:

$$\text{VS}\left(h_k\right)=\begin{bmatrix} S\left(h_k, t_{s,k}, t'_{\text{entry},k}\right) \\ S\left(h_k, t_{s,k}, t'_{\text{entry},k}+\Delta t\right) \\ \cdots \\ S\left(h_k, t_{s,k}, t'_{\text{entry},k}+(V-1)\Delta t\right) \end{bmatrix}_{V\times N\times M} \tag{9}$$

The own occupancy matrix and video sequence are then stacked to form a tensor of four channels, resulting in $(V+1)\times N\times M$ tensor that we refer to as ASOG. ASOG is mathematically defined as:

$$ASOG\left(h_k\right)=\begin{bmatrix} S\left(h_k, t_s\right) \\ S\left(h_k, t_{s,k}, t'_{\text{entry},k}\right) \\ S\left(h_k, t_{s,k}, t'_{\text{entry},k}+\Delta t\right) \\ \cdots \\ S\left(h_k, t_{s,k}, t'_{\text{entry},k}+(V-1)\Delta t\right) \end{bmatrix}_{(V+1)\times N\times M} \tag{10}$$

D. Machine Learning Models utilising ASOG

We propose two different approaches on how to handle the 3-dimensional ASOG in machine learning models.

- The first approach is the application of a 2-Dimensional Convolutional Neural Network (CNN) layer to the 3-dimensional ASOG.
- The second approach is the flattening of the 3-D ASOG feature and handling it as an additional set of tabular features. In this work flattening describes the process of concatenating all tiles from each channel.

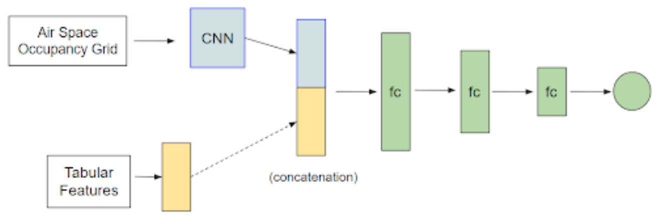


Fig. 5. Concat - Conv NN model

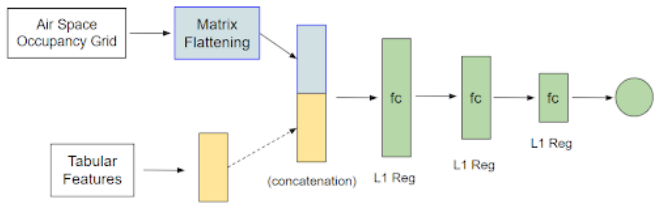


Fig. 6. Concat - Flat NN model

We believe that it is ideal for models to incorporate ASOG and the tabular features in the training process. This is because in such settings, available information is used optimally for training the binary classification model. We propose two different operations to combine the two inputs:

a) Concatenation: The flattened version of ASOG can be directly concatenated with the tabular features since both input types are one dimensional. This concatenation can serve as input for Neural Networks as well as for Gradient Boosted Regression Tree models. If convolution is applied to ASOG, the output of the convolutional layer stack is flattened and then concatenated with the baseline tabular features.

b) Ensemble: Another approach is to combine two separately trained sub models as an Ensemble. We train one model using only ASOG and one model with only tabular features. After both sub-models are fully trained, we find the optimal combination weight a on the validation data. The predictions for the test dataset are generated using the optimal a. Let $\hat{y}_{asog}$ and $\hat{y}_{tab}$ be the output of each model. Then a is searched to maximize the area under the ROC-curve (AUC) of combined prediction y_{ens}:

$$y_{ens} = \begin{cases} 1 & \text{if } (a * y_{tab} + (1-a) * y_{asog}) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

Figure 5 shows the illustration of the NN based model that processes ASOG with CNN and uses concatenation to combine ASOG with the tabular features. We refer to this model as **Concat - Conv NN**.

Figure 6 shows the illustration of the model we refer to as **Concat - Flat NN**. In this model the ASOG does not pass through the CNN layer. Instead, it is just flattened. The flattened ASOG is concatenated with the tabular features before passing through the fully connected layers. L1 regularization "L1 Reg" is applied on the fully connected layers to induce sparsity.

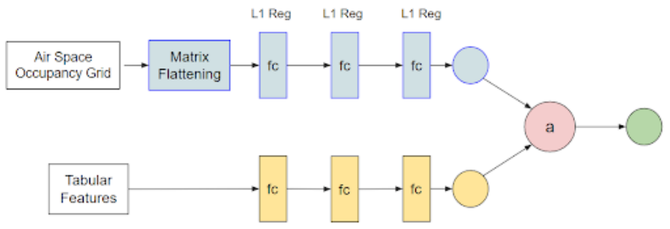


Fig. 7. Ensemble - Flat NN model

Figure 7 shows the illustration of an ensemble model, that we refer to as **Ensemble - Flat NN**. Two separate models are trained. One sub-model is the Baseline model as illustrated in figure 1 and the other is composed of neural networks with L1 regularization that takes in only the flattened ASOG. The **Ensemble - Conv NN** model is similar to Ensemble - Flat NN, but ASOG is fed through a CNN layer, instead of flattened. Concatenation and Ensemble model architectures that are based on XGBoost are also trained.

V. Experiment Details

We set up an experiment to identify whether ASOG improves the performance of the model that predicts if an aircraft deviates from the planned entry point. We will first elaborate on the data being used and the relevant pre-processing done before the training of the prediction model. Finally, we will explain the procedures employed during model training. The implementation of this work is made publicly available in a Github code repository[1].

A. Data

The data used in this work was published by Nilsson and Unger (2022) [26]. The dataset is called Swedish Civil Air Traffic Control (SCAT). It contains close to 170,000 flights, weather forecasts and airspace data collected from the air traffic control system in the Swedish flight information region. The data is divided into 13 distinct weeks that span a year and it is restricted to scheduled flights, with military and private aircraft excluded from the recorded data.

The actual trajectory for each flight is defined by a list of points that include the exact coordinates of a flight together with a timestamp, as shown in (1). A flight plan, which is also referred to as planned trajectory, is defined following that scheme, and therefore, it contains a list of coordinates with time points that indicate when the aircraft is supposed to be at such location, as seen in (2).

From time to time, the flight plan is adjusted to react to changes of circumstances such as weather or potential conflicts with other aircraft, such that conflicts are avoided. This is reflected in the SCAT dataset, such that every flight has a list of flight plans. Each item in the list contains the adjusted flight plan and the time when the adjustment was issued.

[1]The implementation of this work is made publicly available in https://github.com/1503-firmansyah-indra/flight-entry-deviation

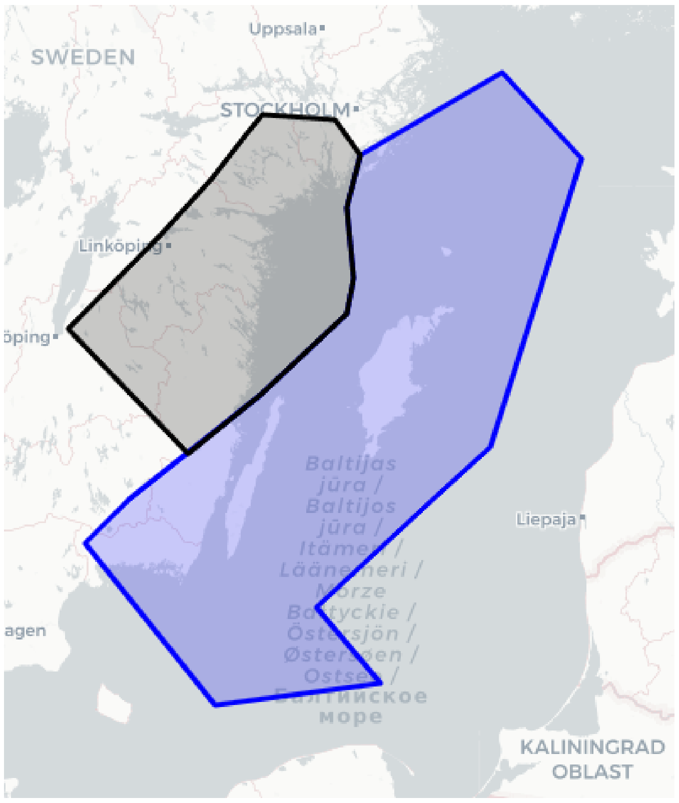

SWEDEN
Uppsala
STOCKHOLM
Linköping
öping
agen
Baltijas jūra / Baltijos jūra / Itämeri / Läänemeri / Morze Bałtyckie / Östersjön / Østersøen / Ostsee / Балтийское море
Liepaja
KALININGRAD OBLAST

Fig. 8. airspace sectors studied in this work. The airspace sector covered in black shade is Sector W, while the one in blue shade is Sector 67Y. Image generated using [25]

Since trajectories are defined by a set of points, the complete trajectory can only be inferred. We do that by assuming that aircraft travel in a straight line between the points. To compute the position of an aircraft during a certain time point, we have to additionally assume that aircraft travel with constant velocity in between points.

B. *Preprocessing*

1) Filtering Relevant Flights: Two Swedish airspace sectors are selected for this work. The first airspace sector being studied in this work is a combination of 3 airspace sectors; ESMM ACC Sector 6, ESMM ACC Sector 7 and ESMM ACC Sector Y; which will be referred to as Sector 67Y. The second sector is ESMM ACC Sector W, which will be referred to as Sector W. A map of these airspace sectors can be found in figure 8. Only flights that enter either one or both of the sectors are used to train the prediction model.

2) Training Data Label Assignment: For each filtered flight, the geodistance deviation from planned trajectory at the sector entry point is calculated using the formula in (3). The calculation is based on the most recent version of the flight plan at prediction time, t_s. In this work, we set t_s as 15 minutes before the forecasted entry time. Some data instances do not have a flight plan update which is older than 15 minutes before the forecasted entry time, and these instances are therefore excluded from the experiment. We use a threshold of 5 kilometers to define the binary deviation classification problem. This means that we set δ in (4) to 5.

3) Tabular Features: In section IV-A, we know that there are already some readily available tabular features. We hypothesize that these tabular features are helpful for training the binary classification model. As mentioned in section V-B2, prediction time, t_s, is set as 15 minutes before the forecasted entry time.

4) Constructing Air Space Occupancy Grid: ASOG consists of two types of matrices that are hypothesized to improve the binary classification model. The dimension of each of these matrices is set to be $\mathbb{N}^{10\times 10}$. One of them is called ”own occupancy matrix”, as conveyed in (6) and (7). The own occupancy has a 1 on every grid that will be crossed following the planned trajectory of the flight.

The second matrix is referred to as ”video sequence”, as conveyed in (8) and (9). The video sequence matrices contain information about which grids are already occupied by other aircraft at different time points; at forecasted entry time, 5 minutes and 10 minutes afterwards. Since 3 different time points are used, video sequence matrices manifest as three two-dimensional matrices. The forecasted entry time is computed based on the latest flight plan at prediction time, t_s. For all other flights in the airspace we compute their position at the 3 reference time points, based on their oldest flight plan version. With that, we determine how many aircraft are in each section of the grid.

The concatenation of the ”own occupancy matrix” ($\mathbb{N}^{10x10x1}$) and the ”video sequence” ($\mathbb{N}^{10x10x3}$) results in the ASOG vector with a dimensionality of ($\mathbb{N}^{10x10x4}$)

We experiment with different ways of dealing with the matrices in ASOG. One approach is to feed the four matrices into a CNN. Another approach is flattening the matrices into features where each feature relates to one element in the 3-D ASOG vector.

C. *Model Training*

There are 12,431 and 20,141 flights in the datasets for sector 67Y and W, respectively. The dataset is imbalanced in both sectors. This means the proportion of instances with deviations is smaller. The imbalance is expected because frequent deviations may suggest that there is a systematic error in which trajectories are planned. We assume that if there is such a systematic error, it will have been identified and resolved. In 67Y, 20.8 percent of the flights have deviation, while it is 16.8 percent in sector W. During training, the dataset is split into three - training (68%), validation (12%) and test (20%) sets. The splitting is done in a stratified manner, such that each data set has approximately the same proportion of instances with positive and negative labels. Instances with positive labels refer to deviating flights. Before model training, oversampling of instances with deviation is done in the training set in order to account for the imbalance.

Experiments are carried out to study the performance of the model architectures, proposed in section IV. The loss function used in the training is binary cross entropy. The detailed implementation and hyper-parameters of the models shared in this section can be found in the code.

In addition to the proposed neural network based approaches, we also train XGBoost (XGB) based models. **Baseline XGB** model is trained by only using the tabular features, while **Concat - Flat XGB** model is trained using the concatenation of the flattened ASOG and the tabular features. The **Ensemble - Flat XGB** utilizes the proposed ensemble method, with two separately trained XGBoost models trained on only tabular features and only ASOG, respectively.

The model training and observation reading are repeated 10 times. In each repeat, a different random seed is set at the beginning. Setting a random seed ensures that we can reproduce the outcomes of processes that involve randomness, such as the dataset splitting and neural network weights initialization. For each model, mean and standard deviation are calculated from the 10 observations.

VI. Results

We conducted experiments to study if the use of ASOG improves the performance of the model that predicts if a flight will deviate from its latest planned trajectory when entering an airspace sector. Tables I and II exhibit the performance of our approaches measured in area under the ROC Curve (AUC).

To understand what AUC metric signifies, one has to understand what Receiver Operating Curve (ROC) is. ROC plots the True Positive Rate (TPR), shown in (12), on y-axis and False Positive Rate (FPR), defined in (13), on the x-axis. AUC metric measures the ability of a binary classification model to separate positive and negative instances. In this work, a positive instance refers to a flight which deviates from the latest planned trajectory when entering an airspace sector. AUC values ranges from 0 to 1. When AUC value is 0.5, the binary classifier is as good as random chance, while AUC value of 1 represents a binary classifier that can perfectly distinguish positive and negative instances.

$$TPR = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{12}$$

$$FPR = \frac{FalsePositives}{TrueNegatives + FalsePositives} \tag{13}$$

The Ensemble - Flat XGB model has the best performance for both airspace sectors. In Sector 67Y, Ensemble - Flat XGB achieves AUC of 0.721 while Baseline XGB, the XGBoost model that is trained only using tabular features, achieves AUC of 0.706. In Sector W, Ensemble - Flat XGB achieves AUC of 0.827 while Baseline XGB achieves AUC of 0.740. Similar behaviour is observed in the results of models with Neural Network architecture. In Sector 67Y, Ensemble - Flat NN achieves AUC of 0.691, which is higher than that of Baseline NN, 0.677. In Sector W, Ensemblle - Flat NN achieves AUC of 0.760, and this is higher than the AUC of Baseline NN, 0.741.

TABLE I
Experiment results for Sector 67Y

	AUC	
Model	**XGB**	**NN**
Baseline	0.706 ± 0.009	0.677 ± 0.009
Ensemble - Flat	**0.721** ± 0.025	0.691 ± 0.012
Ensemble - Conv	–	0.687 ± 0.011
Concat - Flat	0.702 ± 0.014	0.668 ± 0.012
Concat - Conv	–	0.673 ± 0.010

TABLE II
Experiment results for Sector W

	AUC	
Model	**XGB**	**NN**
Baseline	0.740 ± 0.006	0.741 ± 0.031
Ensemble - Flat	**0.827** ± 0.011	0.760 ± 0.021
Ensemble - Conv	–	0.747 ± 0.028
Concat - Flat	0.753 ± 0.011	0.674 ± 0.019
Concat - Conv	–	0.656 ± 0.011

These observations show that there are model architectures where the use of ASOG improves the AUC of the binary classification model. This shows that when ASOG is properly utilized during model training, the performance of the binary classification model can be improved. We can conclude that ASOG does provide additional useful information on top of what has been captured by the tabular features.

However, the use of ASOG does not result in improvement of AUC value in some model architectures. In both sectors, Concat-Flat NN and Concat-Conv NN were unable to outperform the Baseline NN model. Although hyperparameter search was performed, the model architectures are possibly still sub-optimal. Further hyperparameter search could be done to identify neural network architectures that can perform better than the Baseline NN model.

All of our models output a value within the range of 0 and 1. Values higher / lower than the prediction threshold are assigned to the positive / negative class. Per default we used a prediction threshold of 0.5 for the values we report. Changing the prediction threshold can affect the precision and recall values. A higher threshold may result in fewer positive predictions but potentially higher precision, while a lower threshold may yield more positive predictions and higher recall but potentially lower precision. In a real world application, the prediction threshold should be modified based on the specific requirements.

VII. Conclusion

Ultimately, our work is the first to propose and empirically evaluate the use of ASOG for predicting the airspace sector entry point deviation of an aircraft. Prediction of entry point deviation is relevant for ATCOs because it gives them more time to consider the deviation into subsequent planning. This helps ACTOs to keep airspace sectors safe and efficient with less time pressure.

The proposed ASOG is a feature that captures relevant information about sectors and demonstrates an improvement in model performance. As we have shared in section VI, the best model that incorporates ASOG outperforms the baseline model when judged based on performance metrics AUC.

Experiments are carried out on different model architectures and the observed performances vary. The best model for this type of task uses XGBoost on an ensemble setting with a flattening operation on the ASOG. Such results are achieved in both of the airspace sectors that are considered. Neural network based model architectures are able to obtain competent results.

Finally, ASOG provides an intuitive framework to capture the dynamics of the airspace and can be useful in various other applications, such as predicting flight delays and flight trajectories. Additionally, incorporating weather data into ASOG could be explored in future work. Addressing the entry point deviation prediction as a regression problem could also be an area of research worth exploring.

References

[1] I. C. A. Organization, *Procedures for Air Navigation Services: Air Traffic Management*, 16th edition. International Civil Aviation Organization, 2016, Doc 4444.

[2] S. Starita, A. K. Strauss, X. Fei, R. Jovanović, N. Ivanov, G. Pavlović, *et al.*, "Air traffic control capacity planning under demand and capacity provision uncertainty," *Transportation Science*, vol. 54, no. 4, pp. 882–896, 2020.

[3] B. Yu, Z. Guo, S. Asian, H. Wang, and G. Chen, "Flight delay prediction for commercial air transport: A deep learning approach," *Transportation Research Part E: Logistics and Transportation Review*, vol. 125, pp. 203–221, 2019.

[4] S.-H. Chung, H.-L. Ma, M. Hansen, and T.-M. Choi, *Data science and analytics in aviation*, 2020.

[5] R. Patriarca, G. Di Gravio, R. Cioponea, and A. Licu, "Democratizing business intelligence and machine learning for air traffic management safety," *Safety Science*, vol. 146, p. 105 530, 2022, ISSN: 0925-7535. DOI: https://doi.org/10.1016/j.ssci.2021.105530. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753521003738.

[6] A. Degas, M. R. Islam, C. Hurter, S. Barua, H. Rahman, M. Poudel, *et al.*, "A survey on artificial intelligence (ai) and explainable ai in air traffic management: Current trends and development with future research trajectory," *Applied Sciences*, vol. 12, no. 3, 2022, ISSN: 2076-3417. DOI: 10.3390/app12031295. [Online]. Available: https://www.mdpi.com/2076-3417/12/3/1295.

[7] D. Spencer, "Applying artificial intellignece techniques to air traffic control automation," *The Lincoln Laboratory Journal*, vol. 2, no. 3, pp. 537–554, 1989.

[8] Y. Guleria, P. Tran, D.-T. Pham, S. Alam, and N. Durand, "A machine learning framework for predicting atc conflict resolution strategies for conformal automation," in *11th SESAR Innovation Days*, 2021.

[9] A. Degas, M. R. Islam, C. Hurter, S. Barua, H. Rahman, M. Poudel, *et al.*, "A survey on artificial intelligence (ai) and explainable ai in air traffic management: Current trends and development with future research trajectory," *Applied Sciences*, vol. 12, no. 3, p. 1295, 2022.

[10] Y. Guleria, Q. Cai, S. Alam, and L. Li, "A multi-agent approach for reactionary delay prediction of flights," *IEEE Access*, vol. 7, pp. 181 565–181 579, 2019.

[11] A. Degas, E. Kaddoum, M.-P. Gleizes, F. Adreit, and A. Rantrua, "Cooperative multi-agent model for collision avoidance applied to air traffic management," *Engineering applications of artificial intelligence*, vol. 102, p. 104 286, 2021.

[12] S. R. Shah, A. Campbell, and A. Campbell, "Analyzing pilot decision-making using predictive modeling," *Proceedings of the ICRAT*, 2018.

[13] R. Dalmau Codina, S. Belkoura, H. Naessens, F. Ballerini, and S. Wagnick, "Improving the predictability of take-off times with machine learning: A case study for the maastricht upper area control centre area of responsibility," in *Proceedings of the 9th SESAR Innovation Days*, 2019, pp. 1–8.

[14] D.-T. Pham, S. Alam, and V. Duong, "An air traffic controller action extraction-prediction model using machine learning approach," *Complexity*, vol. 2020, pp. 1–19, 2020.

[15] Y. Liu, M. Hansen, D. J. Lovell, and M. O. Ball, "Predicting aircraft trajectory choice–a nominal route approach," in *Proc. of the International Conference for Research in Air Transportation*, 2018.

[16] L. Ma and S. Tian, "A hybrid cnn-lstm model for aircraft 4d trajectory prediction," *IEEE Access*, vol. 8, pp. 134 668–134 680, 2020. DOI: 10.1109/ACCESS.2020.3010963.

[17] X. Chen, J. Lu, J. Zhao, Z. Qu, Y. Yang, and J. Xian, "Traffic flow prediction at varied time scales via ensemble empirical mode decomposition and artificial neural network," *Sustainability*, vol. 12, no. 9, p. 3678, 2020.

[18] H. Mopravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proceedings. 1985 IEEE international conference on robotics and automation*, IEEE, vol. 2, 1985, pp. 116–121.

[19] M. Jardin, "Grid-based strategic air traffic conflict detection," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005, p. 5826.

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

[21] S. Alam, H. A. Abbass, and M. Barlow, "Atoms: Air traffic operations and management simulator," *IEEE Transactions on intelligent transportation systems*, vol. 9, no. 2, pp. 209–225, 2008.

[22] H. Liu, Y. Lin, Z. Chen, D. Guo, J. Zhang, and H. Jing, "Research on the air traffic flow prediction using a deep learning approach," *IEEE Access*, vol. 7, pp. 148 019–148 030, 2019.

[23] B. Başpınar, H. Balakrishnan, and E. Koyuncu, "Optimization-based autonomous air traffic control for airspace capacity improvement," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 6, pp. 4814–4830, 2020.

[24] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[25] *Python library folium*, https://github.com/python-visualization/folium. (visited on 04/10/2023).

[26] J. Nilsson, I. Fedorov, and J. Unger, "Scat - swedish civil air traffic control dataset," 2022. DOI: 10.17632/8YN985BWZ5.1. [Online]. Available: https://data.mendeley.com/datasets/8yn985bwz5/1.