

# DAE : Discriminatory Auto-Encoder for multivariate time-series anomaly detection in air transportation

Antoine Chevrot · Alexandre Vernotte · Bruno Legeard

Received: date / Accepted: date

**Abstract** The Automatic Dependent Surveillance-Broadcast protocol is one of the latest compulsory advances in air surveillance. While it supports the tracking of the ever-growing number of aircraft in the air, it also introduces cybersecurity issues that must be mitigated e.g., false data injection attacks where an attacker emits fake surveillance information. The recent data sources and tools available to obtain flight tracking records allow the researchers to create datasets and develop Machine Learning models capable of detecting such anomalies in En-Route trajectories. In this context, we propose a novel multivariate anomaly detection model called Discriminatory Auto-Encoder (DAE). It uses the baseline of a regular LSTM-based auto-encoder but with several decoders, each getting data of a specific flight phase (e.g. climbing, cruising or descending) during its training. To illustrate the DAE's efficiency, an evaluation dataset was created using real-life anomalies as well as realistically crafted ones, with which the DAE as well as three anomaly detection models from the literature were evaluated. Results show that the DAE achieves better results in both accuracy and speed of detection. The dataset, the models implementations and the evaluation results are available in an online repository, thereby enabling replicability and facilitating future experiments.

**Keywords** Anomaly Detection · Multivariate time series · ADS-B · DAE · Air traffic security

---

A. Chevrot, A. Vernotte, B. Legeard  
DISC/FEMTO-ST Institute, UBFC, CNRS  
Besançon, France  
Tel.: +333-81-66-20-87  
E-mail: name.lastname@femto-st.fr

## 1 Introduction

Over the past ten years, air traffic control has faced a growing number of users and the traffic load keeps growing steadily. With an increasingly congested airspace, numerous new issues are appearing such as flight delays. This increases the overall cost of the flights and exacerbates an already existing tendency for air companies to close down<sup>1</sup> in favour of low-cost companies. In another vein, congested airports imply that the planes stay longer in taxiways which is where they are the least efficient (Polishchuk et al., 2019), increasing their fuel consumption as well as their particle emissions (Zhang et al., 2019).

To tackle these new challenges, Air Traffic Control (ATC) needs improved surveillance technologies supporting the constraints in terms of simultaneously handled aircraft as well as overall accuracy. The Automatic Dependent Surveillance-Broadcast (ADS-B) protocol is currently being deployed world-wide in an effort to improve flights management. ADS-B requires participating aircraft to broadcast their information periodically in an encoded message, like a beacon. This technology embodies the shift from independent and non-cooperative surveillance technologies, historically used for aircraft surveillance, to dependent and cooperative technologies. In this new context, ground stations need aircraft to cooperate and are dependent on aircraft's Global Navigation Satellite System (GNSS) capabilities to determine their position.

Nonetheless, ADS-B is not a new protocol. The ICAO (International Civil Aviation Organization) issued a plan

---

<sup>1</sup> <https://www.nbcnews.com/news/world/british-travel-firm-thomas-cook-collapses-stranding-hundreds-thousands-n1057456>

in 2002<sup>2</sup> recognizing ADS-B as an emerging technology for dissemination of aircraft position information. In 2021, ADS-B is now compulsory in most air-spaces but the protocol itself stayed sensibly the same as it was imagined twenty years ago and the security was not in the highest priority. As a result, anyone with the proper equipment can receive and create messages freely. This liberty in both emission and reception make ADS-B vulnerable to spoofing, and more precisely to attacks called FDIA — False Data Injection Attack — which purpose is to create fake surveillance messages respecting conscientiously the protocol in order to dupe the air traffic controllers to believe in an abnormal situation.

Although ADS-B is not the only protocol used for flights tracking – e.g radar technologies –, it is, as of today, a central brick in the means of surveillance used by public air transportation. In this context, there has been a growing interest for conducting research on anomaly detection systems that address these new threats (Strohmeier et al., 2015b). Among the different existing solutions, some are based on Machine Learning (ML) anomaly detection models. These models already find applications in many different domains like power systems (Wang et al., 2018) or sensor networks (Malhotra et al., 2016) and are found quite popular in recent years. One downside of these models is their need for consequent data availability to achieve meaningful results. It is indeed critical for ML researchers to have access to reliable and genuine data sources to train their models. Thankfully, for ADS-B data, the OpenSky Network (Schäfer et al., 2014) is one of the references in terms of accessibility and data history in air transportation, and one can easily obtain surveillance data from almost anywhere on the globe. This access to genuine data and the lacks of anomalous ones in comparison favours one particular architecture of ML model called auto-encoders.

Auto-encoders are unsupervised ML models often used for anomaly detection and can be found in the literature in many different forms. These models use a first network called the encoder which *encodes* the input data into a latent representation which is then *decoded* by a second network called the decoder. The discrepancies between the input data and the output ones are then used to detect anomalies in the original data. They can be coupled with Recurrent Neural Networks - RNN - to address the temporality of the data (Malhotra et al., 2016). Shown to be quite effective, they have already been used in the past to detect different types of anomalies in the ADS-B protocol like

en-route trajectory anomalies (Olive and Basora, 2019) or spoofing attempts (Ying et al.).

This paper presents a novel type of auto-encoder to use for anomaly detection in ADS-B. The main contributions of this work is listed hereafter:

- (i) *The DAE* –Discriminatory auto-encoder–, a novel asymmetric auto-encoder addressing fluctuations in time series. To the best of our knowledge, this is the first time auto-encoders are used with a single encoder connected to several decoders for anomaly detection in time-series.
- (ii) *The full data framework* using existing tools includes the data cleaning, the feature extraction and the data serialization for model training. Emphasis is made on replicability through a code repository publicly accessible.
- (iii) *Realistic and replicable validation scenarios* are created using an alteration tool to experiment with different types of anomalies. It results in a dataset also available online to compare future models and provide a common base for benchmarks and studies.
- (iii) *Experimental results* using the abovementioned validation scenarios to compare the different existing solutions of ML anomaly detection showing that the DAE performs well overall.

To present the model and the different results achieved with it, this paper has been organized as follows:

*Section 2* provides a basis for understanding the ADS-B protocol, an explanation on FDIAs and the risks associated with it. *Section 3* presents previous works done on anomaly detection for the ADS-B with an emphasis on Machine Learning based techniques. *Section 4* introduces the novel anomaly detection model developed in this paper by detailing its architecture. *Section 5* details the process of data gathering and processing to obtain proper training data for the model. *Section 6* presents the evaluation of this paper, showcasing the data used and the different results obtained using different anomaly detection models. Follow some discussions about implementation and caveats in *Section 7*. *Section 8* concludes this paper.

## 2 Background

### 2.1 ADS-B overview

Communication via ADS-B consists of aircraft using a Global Navigation Satellite System (GNSS) to determine their position and broadcasting it periodically without solicitation (a.k.a beacons or squitters), along with other information obtained from on-board systems

<sup>2</sup> [https://www.icao.int/publications/Documents/9750\\_2ed\\_en.pdf](https://www.icao.int/publications/Documents/9750_2ed_en.pdf)

such as altitude, ground speed, aircraft identity, heading, etc. Ground stations pick up on the squitters, process them and send the information out to the ATC system. The ADS-B data link is generally carried on the 1090MHz frequency. ADS-B is therefore a cooperative (aircraft need a transponder) and dependent (on aircraft data) surveillance technology, which constitutes a fundamental change in ATC. It means for instance that not only ground stations with antennas positioned at the right angle and direction can receive position information. Aircraft can now receive squitters from other aircraft, which notably improves cockpit situational awareness as well as collision avoidance.

The introduction of ADS-B also provides controllers with improved situational awareness of aircraft positions in En-Route and TMA (Terminal Control Area) airspaces, and especially in NRAs (Non Radar Areas). It theoretically gives the possibility of applying much smaller separation minima (e.g., from 80NM longitudinal separation to just 5 NM in NRAs) than what is presently used with current procedures (Procedural Separation) (51, 2005). It has a much greater accuracy and update rate, with a smaller latency. The major drawback of the technology lies in its lack of encryption and authentication, which is discussed in the following section.

## 2.2 False Data Injection Attacks

The progressive shift from independent and non-cooperative technologies (PSR/SSR (Skolnik, 2008)) to dependent and cooperative technologies (ADS-B) has created a strong reliance on external entities (aircraft, GNSS) to estimate aircraft state. This reliance, along with the introduction of air-to-ground data links via Modes A/C/S and the broadcast nature of ADS-B, has brought alarming cyber security issues. Extensive research can be found in the literature that discuss these issues (Schäfer et al., 2013; Zhang et al., 2017; Wesson et al., 2014; Strohmeier et al., 2017), stressing that the introduction of ADS-B has enabled a class of attack referred to as *False Data Injection Attacks* (FDIAs).

FDIAs were initially introduced in the domain of wireless sensor networks (Ma, 2008). A wireless sensor network is composed of a set of nodes (i.e. sensors) that send data report to one or several ground stations. Ground stations process the reports to reach a consensus about the current state of the monitored system. A typical scenario consists of an attacker who first penetrates the sensor network, usually by compromising one or several nodes, and thereafter injects false data reports to be delivered to the base stations. This can lead to the production of false alarms, the waste of

valuable network resources, or even physical damage. Active research regarding FDIAs has been conducted in the power sector, mainly against smart grid state estimators (Dan and Sandberg, 2010; Liu et al., 2011). It shows that these attacks may lead to power black-outs but can also disrupt electricity markets (Xie et al., 2010), despite several integrity checks.

FDIAs also exist in the domain of air traffic surveillance. Because surveillance relies on the information provided by aircraft's transponders to ground stations, aircraft transponders are equivalent to nodes from a wireless network, and ground stations are equivalent to base stations. Although in the ATC domain, there is no real effort to penetrate the sensor network, as all communications are unauthenticated and in clear text. Still, performing FDIAs on surveillance communications requires a deep understanding of the system, its protocol(s) and its logic, to covertly alter the surveillance flow. These attacks are much more complex to achieve than e.g., jamming, because the logic of the communication flow must be preserved and the falsified data must appear probable.

The means of the attacker to conduct FDIAs against ADS-B communications have already been detailed in previous work (Strohmeier, 2016; Manesh and Kaabouch, 2017). Considering the attacker has the necessary equipment, they can perform three malicious basic operations:

- (i) *Message injection* which consists of emitting non-legitimate but well-formed ADS-B messages.
- (ii) *Message deletion* which consists of physically deleting targeted legitimate messages using destructive or constructive interference. It should be noted that message deletion may not be mistaken for jamming, as jamming blocks all communications whereas message deletion drops selected messages only.
- (iii) *Message modification* which consists of modifying targeted legitimate messages using overshadowing, bit-flipping or combinations of message deletion and message injection.

One can sense the potential for disaster if one of these operations was to be executed successfully. It is of the utmost importance that none of the scenarios represent a real threat to such a critical infrastructure with human lives on the line. However, because of the inherent properties of the ADS-B protocol, current solutions for securing ADS-B communications are only partial or involve an unbearable cost Strohmeier et al. (2017). Therefore, ATC systems must become robust against FDIAs, i.e. being capable of automatically detecting any tempering with the surveillance communi-

cation flow while being able to maintain the infrastructure in a working state.

### 3 Related Work

Several takes on improving the security of the ADS-B protocol can be found in the literature. These efforts often fall under several categories but for clarity's sake, these have been separated into two main categories here: one grouping technologies like multilateration or encryption to name a few and the other one grouping Machine Learning based techniques.

#### 3.1 Security solutions for ADS-B protocol

Many works on securing the ADS-B protocol using different technologies already exist with different degrees of feasibility. First, multilateration techniques or MLAT can be used to determine an aircraft position based on measures of time of arrivals (TOAs) of radio feeds. Each ADS-B message is timestamped and broadcast by aircraft. If several radars with synchronized clocks and known positions received the same ADS-B feed, then it is possible to calculate the position of the aircraft based on the differences of TOAs. MLAT can be used to detect ADS-B anomalies (Monteiro et al., 2015) and has the advantage to be very accurate. Recent work from Zhao et al. (2020) also use MLAT to improve the ADS-B protocol accuracy as well as increasing the robustness of the surveillance systems. Fute et al. (2019) show experimentally that FDIAs can also be created to attack multilateration systems assuming an organized attacker with several devices to emit fake ADS-B proving that MLAT can have ultimately similar issues as ADS-B w.r.t. FDIAs.

Regarding the use of the physical layer information, Strohmeier et al. (2015a) create an intrusion detection system based on the strength of the signals they received from 2 different sensors. Similarly, Schäfer et al. (2016) use the Doppler shift measurements to verify the En-Route positions of aircraft over time. Using the clocking system of the Mode-S sensors, Leonardi (2019) manages to obtain similar results to multilateration systems regarding on-board anomalies without the hassle of having at least 4 different sensors. Yang et al. (2019) used Machine Learning methods such as Gradient Boosting or Support Vector Machine to successfully flag anomalies on the PHY-layer features of ADS-B.

On another level, several solutions were proposed for encrypting ADS-B. Based on the identity of aircraft, Baek et al. (2017) describe a confidentiality frame-

work to encrypt the ADS-B. Similarly, Cook (2015) uses Public Key Infrastructure (PKI) to try and secure ADS-B but it either suppresses the open characteristics of ADS-B or requires a change in the protocol itself. Further discussion and analysis can be found in a survey by Strohmeier et al. (2015b).

#### 3.2 Machine Learning based anomaly detection techniques

It is quite common for anomaly detection approaches to rely on multiple sources of data that each represents one aspect of the environment it characterizes, in order to find hidden/complex relationships between sources and rely on these to identify abnormal situations. As an example, one can use satellite images to check real positions of aircraft to confront them against received ADS-B data (Kastelic and Pers, 2019).

In the air surveillance domain, there are multiple surveillance mechanisms (ADS-B, but also legacy radar technologies, multilateration, voice communication, etc) working simultaneously so that air traffic controllers get the clearest possible air situation picture. But there are many areas of the world where most mechanisms could not be deployed, and the sole automated technology that could is ADS-B (e.g., in the center of Australia, or anywhere offshore). Therefore, one of the goals of this work is to find out whether ADS-B alone is enough data to train machine learning models toward the detection of false data reports.

Multivariate time series anomaly detection is an active topic in the Machine Learning community. Supervised learning methods (Karam et al., 2020) require the training data to be labeled and thus can only identify anomalies found in said data. As a result, supervised models have limited usability here and unsupervised approaches are preferred.

Several efforts toward securing ADS-B using unsupervised Machine Learning techniques can be found in the literature. Li et al. (2020) use an hidden Markov model to predict hidden states of the ADS-B protocol and uses them to analyze the deviations during attacks to detect them. Its results however did not show any advantages compared to other Machine Learning solutions like Recurrent Neural Network (RNN) based models in terms of accuracy or false positive rate (FPR). Auto-encoders (AE) are a deterministic family of unsupervised machine learning anomaly detection models often used in the latest publications concerning ADS-B security. Often coupled with RNNs like LSTM or GRU (Malhotra et al., 2016), they have shown good accuracy to detect coarse anomalies (Habler and Shabtai, 2018), or more specific behaviours (Olive et al., 2018;

Olive and Basora, 2019). Li et al. (2019) use LSTM-based auto-encoders in a generative adversarial network (GAN) as a mean to avoid an anomaly threshold selection but misses proper metrics like recall or precision to correctly prove the efficiency of this method. Akerman et al. (2019) use similar LSTM-AE along with convolutional networks to provide images of the traffic and the anomalies to improve the user experience of such solution.

A stochastic variation of the auto-encoder called variational auto-encoder (VAE) are shown to be usable on detecting anomalies in time-series like in the works of Park et al. (2018). Unlike a traditional auto-encoder, which maps the input onto a latent vector, a VAE maps the input data into the parameters of a probability distribution, such as the mean and variance of a Gaussian. Applied to the anomaly detection for ADS-B, Luo et al. (2021) uses an LSTM-VAE model coupled to a support vector data description (SVDD) model to automatically generate its anomaly threshold showing good results on similar coarse anomalies introduced by Habler and Shabtai (2018). However, as pointed by Su et al. (2019), simply coupling LSTM and VAE together ignores the temporal dependence for the stochastic variables. It also assumes a gaussian distribution of the z-space of the ADS-B data which can lead to mediocre results depending on the given data.

Compared with the presented models, the approach developed in this paper is a deterministic uneven auto-encoder using a single encoder to create a latent representation of the ADS-B data linked to several decoders, each getting different data chosen thanks to a discriminating feature. This idea was used by Yook et al. (2020) to separate the sound received by speakers placed differently but to the best of our knowledge, was never used in the anomaly detection field. As a result, the latent space created from the single encoder well represents the ADS-B data while the different specialized decoders well capture the information, addressing the variability of the time series over certain period of time, resulting into better detection.

## 4 DAE : Discriminatory auto-encoder

### 4.1 Problem statement

The task of detecting abnormal ADS-B messages falls into the category of anomaly detection in multivariate time-series. A time series contains successive observations which are usually collected at equal-spaced time-stamp. A multivariate time series  $\mathbf{x}$  of length  $\mathbf{N}$  is defined as  $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , where an observation  $\mathbf{x}_t \in \mathbf{x}$  is an  $M$ -dimensional vector at time  $t$  ( $t \leq$

$N$ ), *i.e.*  $\mathbf{x}_t = [x_t^1, x_t^2, \dots, x_t^M]$  such that  $\mathbf{x} \in R^{M \times N}$ . The dimension  $M$  represents the number of features in an observation  $\mathbf{x}_t$ . In the domain of anomaly detection in time series, the goal is to find out if an observation  $\mathbf{x}_t$  is anomalous or not. However, time windows are usually preferred to single observations in order to get a better understanding of the evolution of the data over time. A Time window  $\mathbf{W}_{t-T:t} \in R^{M \times (T+1)}$  is a set of  $T+1$  observations  $\{\mathbf{x}_{t-T}, \mathbf{x}_{t-T+1}, \dots, \mathbf{x}_t\}$  from time  $t-T$  to  $t$ . The goal is then to determine if a particular time window is anomalous or not.

Even though time windows always come from the same time series, some external contextual factors may alter the shape of the time windows over time. For instance, Figure 1 clearly shows that ADS-B time windows created out of a single flight will have significant differences depending on the phases they are taken from. Hence, every time window  $\mathbf{W}_{t-T:t}$  is associated with a discrimination feature  $\mathbf{D}_{t-T:t}$  to address these discrepancies. The goal of this feature is to mark differences between time windows whether it is time wise, nature wise etc. This can be seen as a static feature that is used by the model in the likes of Miebs et al. (2020) but which is not a part of the training per se. For instance the flight phases from which ADS-B time windows are taken are used as discrimination feature in this paper.

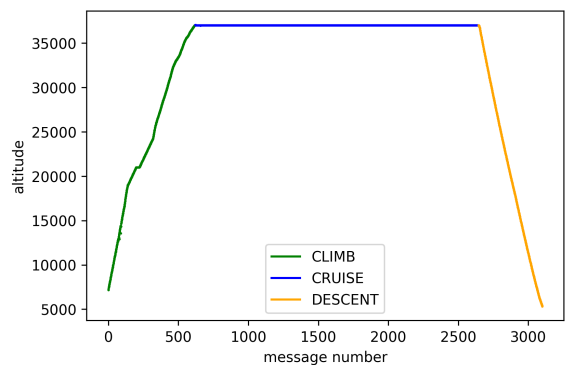


Fig. 1: The 3 different phases of a flight used as discriminating feature

### 4.2 Model architecture

The basis of the DAE model itself uses the architecture of a classic auto-encoder model (Liou et al., 2014) made of an encoder and a decoder. The main difference is its unbalanced numbers of encoder and decoder depending on the discriminating feature presented previously. This

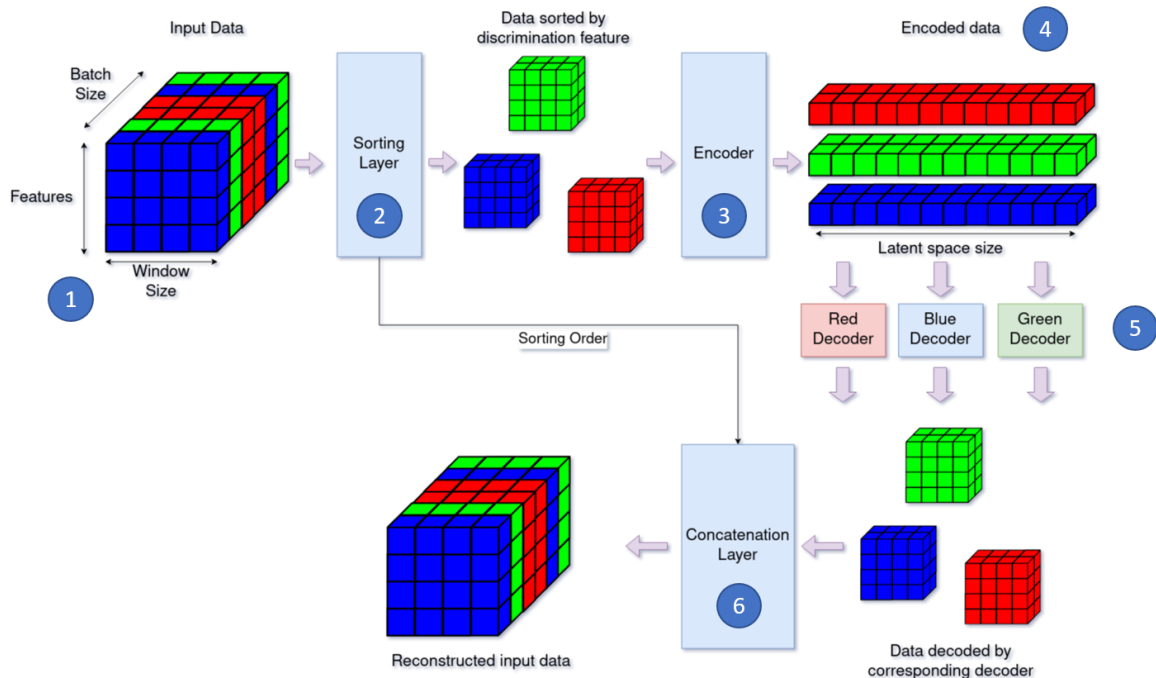


Fig. 2: Architecture of the DAE model

section explains the different parts of the model that can be found in the Figure 2.

① **Input Data:** The input data are constituted of 2 parts. On one hand, the multivariate time-windows are the actual data entering the model. These are 3-dimensional arrays, shaped to be used in RNN layers. On the other hand, the discriminating data is a one-dimensional array which associate each time-window to a discriminant feature.

② **Sorting Layer:** The sorting layer separates the batches of windows into mini-batches according to the discrimination feature. Each mini-batch is then encoded seamlessly. The sorting layer sends the order of the different windows to reconstruct the batch as is to the concatenation layer. The number of mini-batch depends on the number of values the discriminant can take. For instance, the discriminating feature used in the evaluation of this paper is the phase of the flight which is set to 3 (ascending, cruising and descending).

③ **Encoder:** One can use recurrent neural networks (RNN) to address the time aspect of the data. The main problem of classic RNNs is their struggle to learn the long-term dependencies in a sequence because of the gradient vanishing during learning. ADS-B time windows can be up to 60 seconds long and the model must remember what the state of the aircraft was in

this time span. Alternatives to RNNs are LSTMs and more recently the GRU, which do not suffer from the vanishing gradient problem thanks to a system of gating units. In most cases these variants perform equally, and while GRU can have less parameters on smaller dataset, LSTMs having a separate update gate and forget gate can be more effective on longer sequences than the GRU. To add up additional context to the latent representation of the ADS-B time windows, a bidirectional mechanism is added in the encoder layer in order to use both close past and future to encode the data.

④ **Latent Space:** The latent representation captures the normal patterns of the ADS-B multivariate time series, considering their time dependence thanks to the LSTM used in the encoding network. The dimension of this vector is important in the DAE as a small value would likely underfit the input time series while a larger one would increase drastically the training time of the model. The dimension is usually smaller than the original number of features found in the input data but in the case of the DAE, the time dependency itself need to be taken into account, explaining a larger size dimension in the latent space than the input. Precise dimension used during experimentation is showcased in the Section 6

⑤ **Discriminated Decoders:** The decoding part of the DAE is mainly what makes it different from a reg-

ular auto-encoder. While the encoder encodes all the mini-batches yield by the sorting layer seamlessly at the same time, the decoding part is carried out by several decoders, one per mini-batch. This leads to specialized decoders depending on the data they received during their training. Compared to the encoder, the decoders are composed only of a single LSTM layer and not a Bi-LSTM as it was not deemed important for the decoding since the information was already included in the latent representation. As a result, the DAE is asymmetrical in two ways: the numbers of encoders and decoders are different and the encoder is overall bulkier than the decoders. This is justified by the importance of the quality of the encoder knowing it is alone to complete its task.

⑥ **Concatenation Layer:** this layer uses the order kept in memory by the sorting layer to reconstruct the data w.r.t. its original order. It is solely used for the training due to the use of different thresholds for each decoder for the anomaly detection, making this concatenation layer obsolete once training is over.

### 4.3 The thresholds calculation and the anomaly detection

Once the data is reconstructed by the decoders, the input and the output are compared to calculate a similarity score. The higher this score is, the better the model managed to recreate the input time series. Here, each time window  $\mathbf{W}_{t-T:t}$  gets its own reconstruction score calculated by, for instance, a mean squared error. After training, instead of using directly the reconstruction score, an anomaly score is defined as:

$$Anomaly(\mathbf{W}_{t-T:t}) = \frac{1}{n} \sum_{i=0}^n 1 - (\mathbf{x}_i - \hat{\mathbf{x}}_i)^2 \quad (1)$$

This score is compared against a threshold to determine if the window associated to it contains an anomaly or not. As discussed in the previous section, the model has different decoders trained on different data. As a result, the loss of each decoder is going to be different, leading to anomaly scores not being equivalent across the different decoders. In this context, having the same anomaly threshold for all the decoders would be counterproductive and lead to mediocre metrics.

Calculating the threshold can be done in several ways. Luo et al. (2021) uses support vector data description (SVDD) to determine automatically the best threshold to use. SVDD is an unsupervised model that creates boundaries around the training dataset that is then used on testing data to determine whether they are out of bounds (i.e. anomalous). It is usually trained

using a mix of positive and negative to make it more robust to outliers contained in the training set. Unfortunately, in the case of ADS-B, not only real life anomalous examples are scarce, but the data also tend to contain outliers due to already discussed problems which make an SVDD hard to use successfully on real not-over-processed data.

For simplicity and efficiency, the 3-sigma rule is used to calculate the threshold for the DAE. Considering each decoder has its own output distribution, the calculation for the threshold is done on the training data for each one of them. The threshold  $\tau$  is defined as  $\tau = \mu + 3\sigma$  where  $\mu$  is the mean of the anomaly score distribution of one decoder and  $\sigma$  is its standard variation. This results in having a threshold value being different depending on the decoder the data went in.

The distribution of the anomaly scores, which roughly following a normal distribution (see Figure 3), assures the 3-sigma rule to yield a low false positive rate while being sensitive enough to flag anomalies.

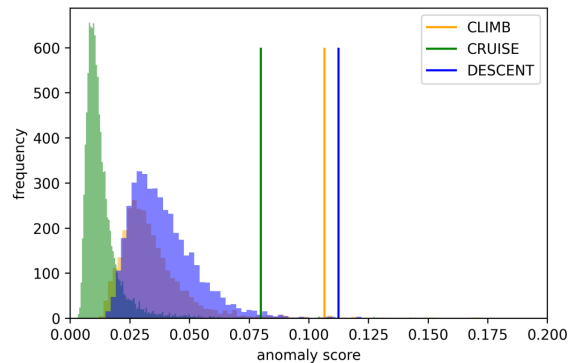


Fig. 3: The anomaly score’s distributions of the training data for each phase along with its calculated thresholds

## 5 Dataset creation and pre-processing

Apart from the DAE, another contribution of this paper is the availability of a dataset to train and evaluate multivariate anomaly detection models. This section presents the different tools used for the creation of the dataset as well as the pre-processing to obtain the final data.

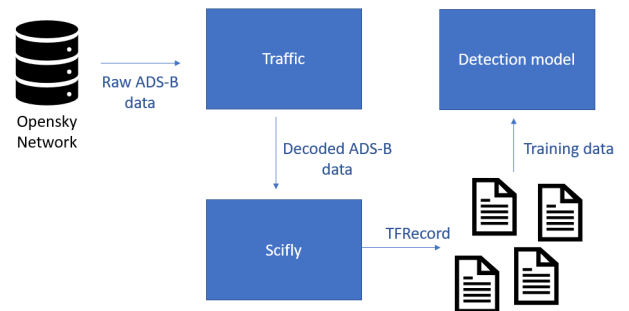


Fig. 4: The data architecture used for gathering and processing ADS-B messages into training data.

### 5.1 Global architecture

In Figure 4 is described the overall architecture used to train the DAE. After retrieving data from the Open-sky Network, the data pre-processing cleans the data, getting rid of aberrations caused by decoding errors or sensor inaccuracy and then creates the different features needed for training. Through serialization, the processed data are then sent to the Model training block which creates the windows and standardizes the data inside the Tensorflow’s data.io framework to stream the data directly during the training using tfrecord format. The model learns the normal patterns of flights through regular historical data and outputs an anomaly score for each time windows. These normal anomaly scores are then used to determine a threshold which separate normal data from anomalous ones using the 3-sigma rule. The model can then be used on unseen data to determine their nature by calculating its score. If the score is under the threshold, then the time window is considered normal, else, it is considered abnormal.

### 5.2 Data acquisition

① **Opensky Network** is an online flight tracking network which provide access to data collected by cooperative ground stations. The large-scale dataset of this evaluation is historical data extracted from their historical database.

② **Traffic** (Olive, 2019) is an open-source Python-based tool allowing users to query the Opensky Network historical database. It simplifies the data gathering by aggregating the different types of ADS-B messages (position, velocity and identification) as well as making the data cleaning less cumbersome. The Opensky Network data comes in two main different forms available to the user : one in a form of already processed and cleaned data called state-vectors and the other in raw messages in BEAST format. While the former would be a time-saver, it would not yield full control over the data preparation. On the other hand, getting the raw transmission not only gives more freedom to the user but also allows experiments using directly an ADS-B feed delivered by a private antenna. With recent iterations of Traffic, the processing of raw data has become much easier. With the implementation of a clustering algorithm used by Sun et al. (2017), the raw data which

consist of series of messages, once decoded, can be separated into well defined flights.

As mentioned earlier, the relevant ADS-B messages are sent from position, velocity and identification messages. These messages are, according to the ADS-B specification, sent by the transponder every half-second for the position and velocity messages and once every five seconds for the identification ones. This very short timeframe between receptions leads to a consequent amount of data with a non-negligible redundancy. Traffic allows for the downsampling and the concatenation of the different messages. The original timeseries are then transformed from several messages per seconds down to one every two seconds. This has the clear advantage of reducing the size of the dataset without losing meaningful data due to the high redundancy implied by the high-rated emissions. Another valuable gain from this reduction of messages is the amount of information contained into a time window. Indeed, without the re-sampling, a time window of 30 messages would be equivalent to around 13 seconds of recording. Changing the original rate of messages to 2 seconds bring the 30 messages window to a full minute of recording. This strongly impacts both the training time of the model as well as its accuracy as it improves the time dependencies developed by the RNN layers.

③ **Scifly** is a toolbox additional to Traffic<sup>3</sup> developed in the context of the current work. Despite using data from a well-covered area like Europe, errors in decoding the data or approximation from sensors still happen which often result in big leaps of the aircraft during a flight. These corrupted data are undesired in training dataset and would result in lower quality models. To filter out these blatant outliers, we check the distance in kilometers between close neighbour messages (consecutive ones) and separated messages.

The other use of Scifly is to export the ADS-B data in TFRecord. TFRecord format uses protocol buffers<sup>4</sup> to serialize data making it available to a large share of machine learning algorithms. It also have the advantage to allow to shard the data in multiple files to parallelize the input data for optimize training. Lastly, Scifly allows the exporting and importing of data to and from FDI-T, our anomaly creation platform.

### 5.3 FDI-T

*FDI-T* is a testing framework that we developed (Verotte et al., 2021) jointly with Smartesting (<https://www.smartesting.com>) and Kereval (<https://www.kereval.com>).

<sup>3</sup> <https://github.com/Wirden/scifly>

<sup>4</sup> <https://developers.google.com/protocol-buffers/>



kereval.com/). It allows ATC experts to design FDIA scenarios to alter (i.e. create, modify and delete) recorded legitimate ADS-B surveillance messages. The altered recordings can then be played back (w.r.t. time requirements) onto real surveillance systems or can be exported e.g. to train and/or validate Machine Learning models. The goal is to simulate an attacker tampering with the surveillance communication flow.

The types of alteration to apply are specified through the definition of alteration scenarios, of which the design is textual-based via a Domain Specific Language (DSL). Once designed, the scenarios are automatically applied on source recordings of air traffic surveillance communications, thanks to a dedicated alteration engine (Cretin et al., 2020). Alteration scenarios have various parameters, such as a time window, list of targeted aircraft, triggering conditions, and others parameters related to the alteration's type. All parameters are recording agnostic, meaning that scenarios can be applied to multiple recordings regardless of their nature. All these features truly make the creation of ML dataset a fast albeit precise procedure.

Concretely, FDI-T was used in this study to create many of the scenarios that constitute the evaluation dataset.

#### 5.4 Training data

The different flight routes used for the training can be visualized in Table 1. It compiles together 15 flight routes for a total of 1008 flights. The training dataset is exclusively focused on internal European flights. This choice is motivated, mainly, by the excellent land coverage of the OpenSky Network in this area. This ensures good quality data without major discrepancies due to low quality ground station or non-covered area. The dataset is composed of both long and short flights, as well as flights traveling in different directions to ensure data diversity. From the data gathered through the described architecture, only some features of ADS-B messages are kept and fed to the model:

- **Altitude** in feet given by the airborne position messages.
- **Consecutive Delta** in kilometers. This is the Vincenty distance between two consecutive messages calculated from the latitudes and longitudes. This distance is bound to change from 2 main factors. The first one is the change of speed of the aircraft and the second one is the absence of messages picked up by the OpenSky Network. The third reason would be errors in decoding or from sensor malfunctions

but most are filtered out from the data cleaning processing explained above.

- **Tracking Delta**. Difference between the tracking received through ADS-B and the *ideal* tracking calculated from the position of the aircraft and the position of the arrival airport.
- **Vertical Rate** in feet/mn. Represents the aircraft's vertical speed – the positive or negative rate of altitude change with respect to time.
- **Groundspeed** in knots. Represents the speed over ground.
- **Phases**. Categorical feature used as the discriminating feature to choose the decoder. The fuzzy logic developed by Sun et al. (2017) is used to automatically determine the phase a window is originated from. In addition, a rule has been added forcing the cruising phase when the flight is over 300 kilometers away from the departure or arrival airport. This helps when the fuzzy logic labels a crash as a simple descending maneuver. Figure 1 shows the different phases of a flight automatically determined by the Scify algorithm.

It is worth noting that some base features of ADS-B like the tracking, the latitude and the longitude are not directly used in the dataset. Concerning the tracking, the feature being a cyclic feature in degree, experiments were made using the sine and cosine component to avoid the discontinuity implied by having a heading varying between 0 and 360 when 0 and 360 being de facto the same angle. Unfortunately, having two features for the heading instead of one doubled its impact on the model and created some unbalance hence the choice for the heading delta feature presented earlier.

In a similar fashion, the latitude and longitude also being cyclical data were turned into the consecutive delta feature. Another reason for this change is the will to make the model area-agnostic which would have been impossible with the coordinates as is as features. This will improve the accuracy of the models on data they have not seen during their training, as shown by Fried and Last (2021).

#### 5.5 Evaluation Data

The evaluation dataset is composed of different scenarios to test different situations an anomaly detection model could be used in. It is a mix of regular data taken from real life and anomalies created or not using the FDI-T framework. All the anomalies, except the one already found in real data, are applied on all the flights found in the training dataset found in Table ?? from January 2021 instead of the last 3 months of 2020:

Departure airport	Arrival airport	Number of flights	Duration (hours)
Athens (LGAV)	London (EGGW)	56	3.6
Berlin (EDDB)	Kiev (UKBB)	33	1.6
Budapest (LHBP)	Dublin (EIDW)	43	2.8
Frankfurt (EDDF)	Lisbon (LPPT)	68	2.5
Hamburg (EDDH)	Barcelona (LEBL)	29	2.0
Kiev (UKBB)	Paris (LFPG)	83	3.3
London (EGGW)	Milan (LIMC)	46	1.6
Madrid (LEMD)	Moscow (UUEE)	59	4.2
Malaga (LEMG)	Frankfurt (EDDF)	81	2.9
Manchester (EGCC)	Istanbul (LTFJ)	75	3.8
Munich (EDDM)	Lisbon (LPPT)	68	3.3
Paris (LFPG)	Oslo (ENGM)	34	1.9
Stockholm (ESSA)	Barcelona (LEBL)	25	3.2
Vienna (LOWW)	Copenhagen (EKCH)	83	1.3
Zurich (LSZH)	London (EGLL)	225	1.2
Hamburg (EDHI)	Hawarden (EGNR)	45	1.5
London (EGLL)	Moscow (UUEE)	184	4.0

Table 1: Flights used for the training of the different models presented. 15 flight routes data taken from September to December 2020 for training and 2 flight routes taken in January 2021 for validation.

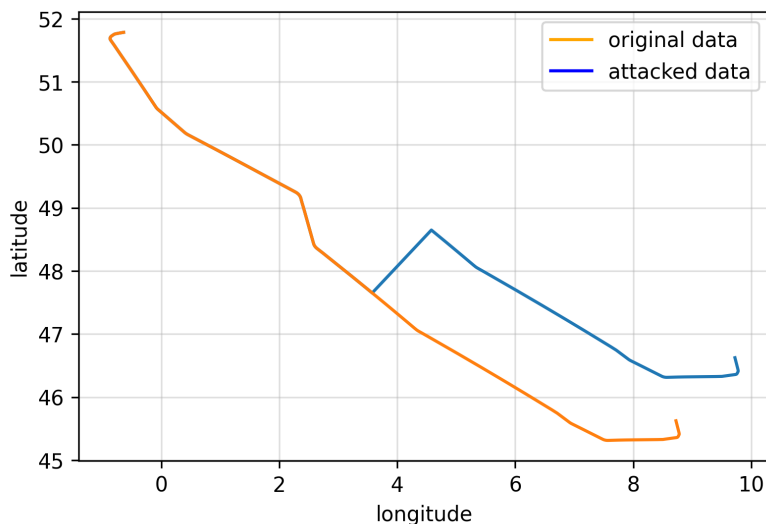


Fig. 5: Constant position offset attack

**World Data (WORLD)** – As the training dataset is exclusively composed of data from European flights, including regular data from other parts of the world in the testing dataset allows for checking the genericity of the approach. It includes flights from the european airspace, american airspace – e.g. Dallas to Louisville – or australian airspace – e.g. Camberra to Perth –.

**Gradual Drift (DRIFT)** – Anomalies that consist of simulating an altitude drift or a velocity drift. The altitude or velocity messages on the attacked time window are all raised/lowered by an increasing/decreasing multiple of  $n$  feet. So, if the first message is lowered by

25 feet, the second will be lowered by 50, etc. The Figure 6 shows a velocity drift used during the evaluation

**Made-up Crashes (CRASH)** – Using FDI-T, life-like crashes scenarios can be created combining an altitude drift, a negative vertical rate, and a reduction of groundspeed – not to be mistaken with airspeed –. The signal is then stopped once the aircraft lands. Figure 7 shows some of the features modified during a CRASH attack.

**Ryanair Hijack (HJK)** – Constituted of the Ryanair flight 4978 from Athens to Vilnius which was forcibly diverted to Minsk after entering the Belarus airspace

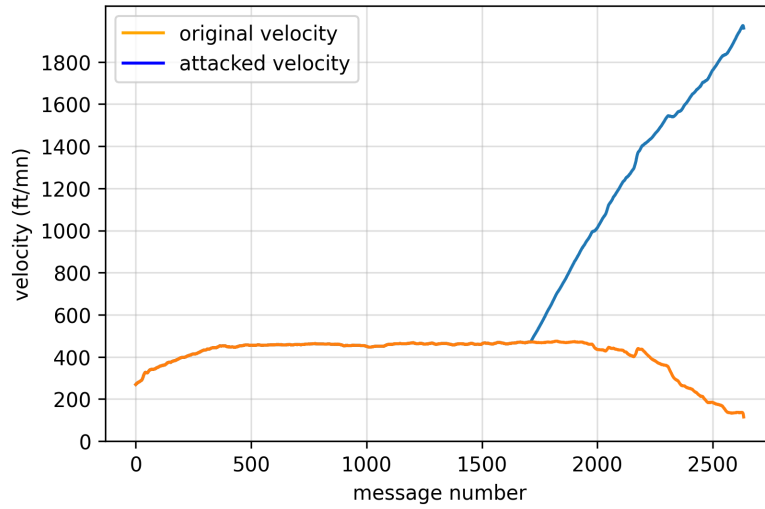


Fig. 6: Velocity drift attack

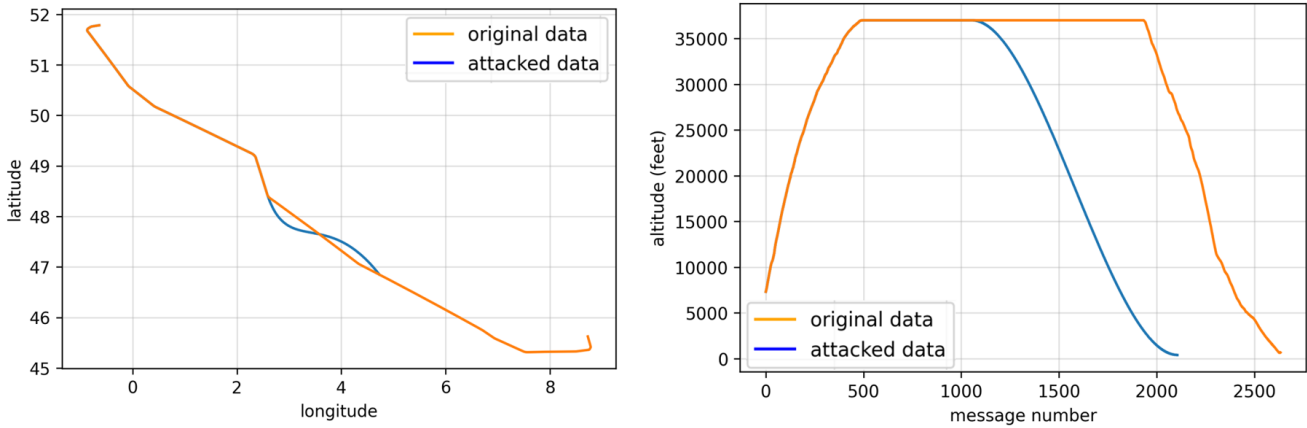


Fig. 7: Crash attack. Latitude / longitude on the left and the altitude on the right.  
Other features like vertical speed or track are also modified *realistically*.

on the 23rd of May 2021<sup>5</sup>. It is to be noted that the emergency was turned on by the crew 2 minutes after the flight started to change its course. For the evaluation, the labels have been set to 1 from the beginning of the emergency till the landing.

**Constant position offset (OFFSET)** – This anomaly, when toggled takes the real data of a flight and adds an offset of 1 in both the latitude and the longitude (see Figure 5). This offset represents a distance of around 132 kilometers between the original and the anomalous trajectory.

## 6 Experimental Evaluation

This section presents the experimental evaluation of the model presented above. It also displays the differences in performance between the DAE and other anomaly detection methods.

### 6.1 Model training specifications

The DAE model was trained with the training dataset above-mentioned which represents 336 Mo of data separated into 15 tfrecord files. Tensorflow interleaves the data contained in these files to feed it to the model during the training, avoiding risks of memory overflows. The training was made on the Mésocentre de Calcul de Franche-Comté using a Tesla V100 performing at

<sup>5</sup> <https://www.flightradar24.com/blog/ryanair-flight-4978-to-vilnius-forcibly-diverted-to-minsk/>

7.8 TeraFLOPS. The training on average was taking around 26 minutes per epoch. In Figure 8 a difference in loss between the training set and the validation set can be observed. It is due to a few outliers in the training set which make the average training loss way higher than its validation counterpart.

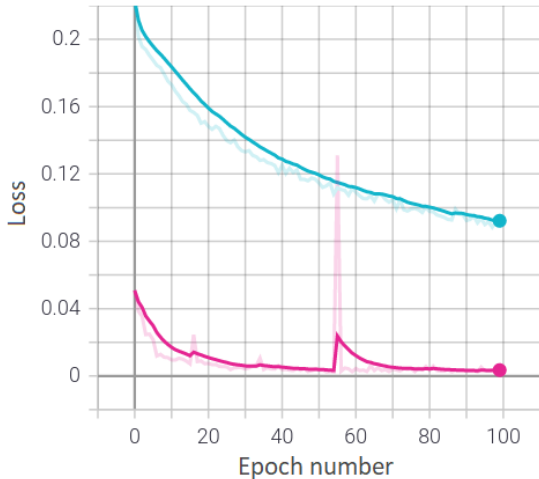


Fig. 8: The training loss per epoch. In blue is the training loss, in pink the validation loss

To train the DAE of which the results can be found in the following section, windows of 30 messages have been used. The batch-size is 256. The number of units in the encoder’s BiLSTM is 32 which is then flattened to feed a Dense layer reducing the dimension to 10, the chosen latent space size. The different decoders each embed a single LSTM layer with 32 units. For the other models found in the evaluation, the features, the hyper-parameters and the threshold selection method found in their respective papers were used if provided.

In order to properly evaluate and compare the different models performance, accuracy (ACC), Recall (R), False Positive Rate (FPR) and F1-score (F1) are used:

$$\begin{cases} Acc = \frac{TP+TN}{TP+TN+FP+FN} \\ R = \frac{TP}{TP+FN} \\ FPR = \frac{FP}{FP+TN} \\ F_1 = \frac{2TP}{2TP+FP+FN} \end{cases}$$

where TP, FP, FN and TN being the values found in a regular 2x2 contingency table.

All the results and implementation of this paper are accessible on the Scifly <sup>6</sup> Github repository. The full dataset used for the different models can also be found at this address. This is made as an effort to improve the replicability of the presented evaluation as well as proposing a non-exhaustive, upgradable base-

line dataset for future models in the growing field of anomaly detection in ADS-B data.

## 6.2 Compared Results against other ML methods

To show the overall performance of the DAE, it is compared with 3 other unsupervised approaches for anomaly detection in ADS-B time-series : a regular Isolation Forest (Liu et al., 2008) model, an LSTM-auto-encoder (Habler and Shabtai, 2018), and a VAE-SVDD (Luo et al., 2021). Table 2 shows the accuracy, the recall, the FPR and the F1 score on the different dataset for each model. For the VAE-SVDD, the method to choose the anomaly thresholds is already given in the paper and the F1 score is calculated accordingly. For the other models, the 3-sigma ruled is applied on the training data to choose the threshold meaning that approximately 99.7% of the training data anomaly score are under this value. Overall, these experimentation results demonstrate the superiority of the DAE compared with the state-of-the-art approaches in ADS-B anomaly detection. Indeed the F1 score on the Total Dataset is more than 20% over the second best performing model (not considering the IForest for the reasons explained later). It is to be noted that the WORLD dataset does not have any true positives nor false negatives which automatically set the Recall to Nan (division by zero) and the F1 to 0. Next, we analyze the performance of the different methods in detail.

*LSTM-AE* is a sequence to sequence model based on a encoder-decoder reconstruction used by Habler and Shabtai (2018) for anomaly detection in ADS-B time-series. This simple deterministic model well manages to capture the ADS-B normal behaviour in its latent space showing very low FPR using a 3-sigma threshold as well as decent results on the Velocity Drift dataset. Its very low F1 score on the Made-up Crash dataset can be explained by the data resembling a regular descent trajectory which leads the decoder to reconstruct the data as is. Lowering the threshold to a 2-sigma could help raising the F1 score but would result to a FPR being way too high for anomaly detection. From the metrics alone, the DAE seems to be under-performing compared to the LSTM-AE for the hijack anomaly. This can be explained by looking at Figure 9 which compares the anomaly score over the message windows for both models. One can observe that for the DAE, the anomaly is set off before the actual emergency due to its delay with the diversion of the flight. It explains the FPR being way higher than the other models and displays the reactivity of the DAE in such circumstances. On the other hand, the low recall is due to the score going back to a *normal* value after some time which means

<sup>6</sup> <https://github.com/Wirden/scifly>

Evaluation Dataset	Evaluation metric	LSTM-AE	IForest	VAE-SVDD	DAE
World Data	Accuracy	0.994	0.687	0.899	0.989
	Recall	NaN	NaN	NaN	NaN
	FPR	0.006	0.313	0.101	0.011
	F1 score	0	0	0	0
Ryanair Hijack	Accuracy	0.946	0.890	0.722	0.847
	Recall	0.637	1	0.227	0.301
	FPR	0.001	0.129	0.231	0.017
	F1 score	<b>0.778</b>	0.729	0.152	<b>0.439</b>
Velocity drift	Accuracy	0.933	0.944	0.949	0.961
	Recall	0.809	0.957	0.930	0.912
	FPR	0.001	0.063	0.043	0.012
	F1 score	0.886	<b>0.937</b>	0.926	<b>0.939</b>
Constant position offset	Accuracy	0.519	0.709	0.541	0.526
	Recall	0.033	0.491	0.077	0.053
	FPR	0.001	0.073	0.046	0.004
	F1 score	0.060	<b>0.598</b>	0.107	<b>0.097</b>
Made-up Crash	Accuracy	0.506	0.919	0.710	0.962
	Recall	0.003	0.922	0.426	0.929
	FPR	0.001	0.084	0.037	0.004
	F1 score	0.005	<b>0.925</b>	0.573	<b>0.955</b>
Total	Accuracy	0.780	0.830	0.764	0.857
	Recall	0.371	0.843	0.415	0.549
	FPR	0.002	0.132	0.092	0.010
	F1 score	0.544	<b>0.797</b>	0.440	<b>0.738</b>

Table 2: Comparison of the different models evaluated

the DAE does not label the end of the flight as abnormal from its ADS-B data.

*VAE-SVDD* is a variational auto-encoder (VAE) coupled with a support vector data description model (SVDD) to automatically determine its threshold. A VAE is a deep Bayesian model which represents an input  $\mathbf{x}_t$  to a latent representation  $\mathbf{z}_t$  with a reduced dimension, and then reconstructs  $\mathbf{x}_t$  by  $\mathbf{z}_t$ . The main difference with a regular auto-encoder is that the latent variable  $\mathbf{z}_t$  is sampled from a probability distribution, such as a Gaussian distribution with the mean and the standard variation being outputs of the encoder network. This stochastic approach could explain the better results compared to the regular LSTM-AE on the Made-up Crash and Velocity drift dataset. Luo et al. (2021) combine LSTM and VAE by replacing the feed-forward network in a VAE to GRU but do not include information from  $\mathbf{z}_t - \mathbf{1}$  into  $\mathbf{z}_t$  in the likes of Su et al. (2019). That might explain the issues the VAE-SVDD has to properly represent the distributions of the input data, leading to high FPR compared to the other methods. All in all, the VAE-SVDD, while performing well on coarse anomalies like the velocity drift and to some extents the Made-up Crash thanks to its stochasticity, fails to reconstruct properly ADS-B data leading to high FPR on new data and mediocre results overall. This could be explained by the limitation of having

a Gaussian *qnet* being too simple to properly reconstruct ADS-B information coming from other parts of the world, negating the advantages of having such an architecture.

*Isolation Forest* is an anomaly detection algorithm using an ensemble of isolation trees to differentiate normal data from anomalies. It has the advantage of being fast, light-weight and can be quickly implemented. The model yields good results when compared to the other models, which is explained by the evaluation dataset being based on the same flights as the training data but one month later. The IForest manages to flag anomalies on flights it has already seen or in the vicinity of these said flights – for instance, the Ryanair Hijack – without any trade-off except its FPR. Indeed, the FPR is on average almost ten times higher than the DAE’s which makes it hard to use as a reliable anomaly detector. It would even be completely pointless on flights in part of the world it did not see during its training.

**Summary.** Compared to the LSTM-AE with a single decoder, the DAE, thanks to its specialized decoders, manages to discriminate anomalous situations like crashes from regular descent operations while keeping a very similar low FPR overall. Compared to the VAE-SVDD, the DAE performs better on all dataset except on the constant position offset where all models perform poorly due to the scenario’s very nature.

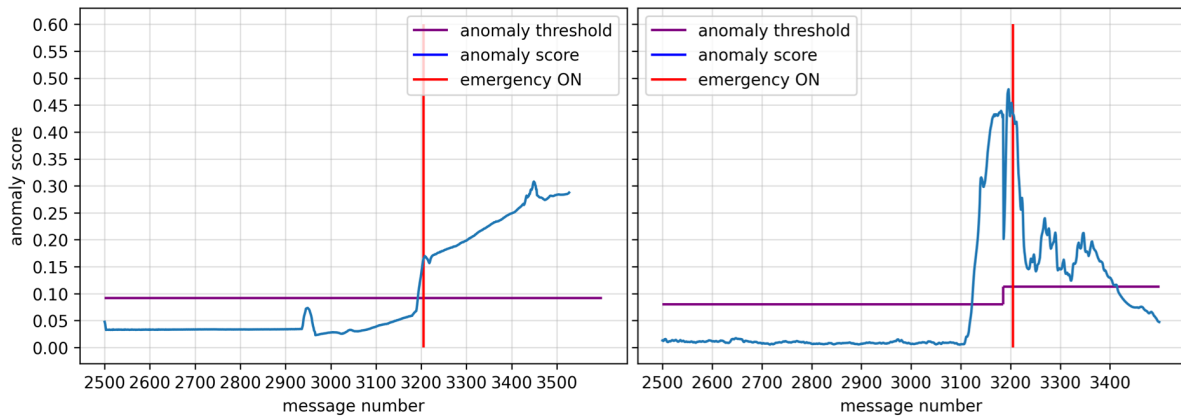


Fig. 9: Anomaly score for the LSTM-AE on the left and the DAE on the right for the Ryanair hijack flight.

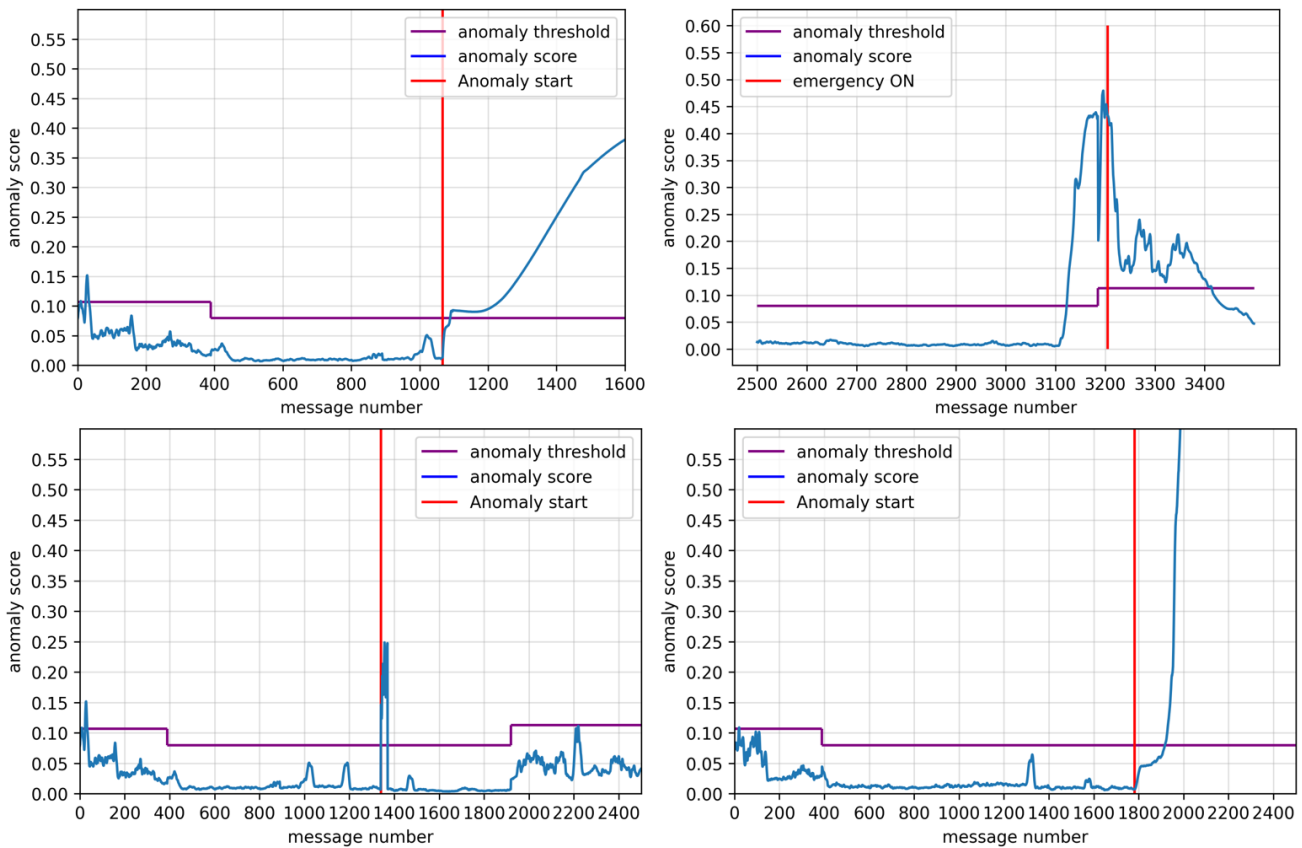


Fig. 10: DAE anomaly scores for a flight taken randomly from the different evaluation dataset. The top-left figure is from the CRASH dataset, the top-right is from the Ryanair hijack, the bottom-left is from the constant position offset and the bottom-right is from the velocity drift dataset

Indeed, the small offset added to the latitude and longitude is not enough to trigger alarms leading to extremely low F1 scores. This anomaly can only be detected by the LSTM or GRU based models when the values are changed. Finally, the IForest model, despite

being cost-effective and accurate on the few flights it has seen during its training, is not as dependable as the LSTM-AE or the DAE, limiting its usage in real-life applications.

## 7 Discussion

The DAE model shows good results on the chosen evaluation dataset compared to other ADS-B anomaly detection models. Here are some discussion points and caveats for using and improving the model in future works:

- The first assumption made for the usability of these models is the authenticity of the data used during the training. If data sources like sensors or the Opensky-Network were to be attacked, the models trained from these corrupted sources would not be able to detect ADS-B anomalies properly.
- Flight trajectories, while being overall linear over the same routes, can have inconsistent trajectories, mainly due to fluctuating weather or congestion problems. This results in ADS-B time-series having a tolerance margin when used to train ML models. This means that all attacks made within this margin will likely end up not being detected if the attacker carefully conforms to the ADS-B protocol and to the flight plan.
- The DAE in its current state does not support online learning and therefore cannot be updated to the latest ADS-B data. However, all the data used in the evaluation dataset are from 2021 while the training data were from 2020 showing no significant differences between them. This result only has two explanations: either the data does not change significantly enough over time to make a difference or the model is robust enough to not be disturbed by small changes. Only future data will give proper insight to answer this. In addition, the low FPR on the world dataset shows that the model is area-agnostic thanks to the features and the data-processing used for the data. This avoid the training of different models for specific regions.
- One of the downside of the creation of *realistic* scenarios through a framework like FDI-T is the introduction of a tool bias which could lead to the detection of anomalies being eased. While this would question a supervised approach being trained using said data, for the unsupervised approach, it only shows that models are able to detect these abnormal scenarios. If coupled with a few real life examples of anomaly situation, it only constitutes contents to prove the robustness of the models.
- The Ryanair anomaly is detected the quickest by the DAE but it is also the model where the anomaly disappear once the main change in track is over. This can be explained by the switch to the DESCENT decoder which is less sensitive to changes in track due to the flight activity when approaching the arrival

airport including congestion management and level flight. These results could be improved by adding other decoders taking level phase data or congestion management data.

- In this experiment, all the decoders of the DAE have the same hyper-parameters and the same architecture. One could decide to make one decoder bulkier or smaller depending on the data fed to it. In the case of the ADS-B, the climbing and the descending data being more complex, it would make sense to have deeper or bigger decoders.
- Having a FPR higher than zero can be a problem in the air traffic management as it would trigger unnecessary measures to take care of false alarms. Unfortunately, it is not easy task to create a model sensitive enough to detect all kind of anomaly without ever have false positives. On Figure 10, the false positives observed barely exceed the threshold while the anomaly scores like the one on the Ryanair hijack will almost reach five times the threshold value. Adding other *soft* thresholds – e.g. four or five sigma rule – to determine the gravity of the anomaly could help discarding the false alarms in most cases and on the other hand could raise emergencies if the anomaly score would go too high, disregarding entirely the rest of the flight. This strategy would help in the case of anomaly spikes like in the constant position offset.

## 8 Conclusions and Future Work

Detecting anomalies in the ADS-B protocol can greatly improve the monitoring and troubleshooting of the airspace for the air traffic managers in a timely manner. In this paper, we introduced the DAE, a novel auto-encoder architecture to detect anomalies in ADS-B multivariate time-series which work efficiently in any ADS-B covered area, with low chances of false alarms. Thanks to a complete data acquisition framework and tools available online, a baseline dataset was created to train, validate and evaluate machine learning models implementation, also fully open-accessible. The results presented on this dataset show that the DAE model perform as well or better than other comparable anomaly detection models and can be more reactive on emergencies.

Potential future work following these results can be separated in two main areas:

- First, strong beliefs upon the re-usability of this architecture will be audited in the ATC domain using other discriminating features like the type of aircraft or the kind of sensors used to gather the ADS-B data. Other data could also complement the current

ADS-B data to improve the accuracy of anomaly detection models like the DAE. Weather data could be added to better isolate storm avoidance manoeuvres from anomalous situations. The use of the COMMB data, which are information also broadcast by the aircraft, could also help to determine the authenticity of the ADS-B. The use of these other forms of data will require data collection efforts as they are not as easily accessible as the ADS-B.

- In order to validate the DAE approach, there is a need to check its efficacy in other domains. The maritime domain uses the AIS protocol, very similar to ADS-B, with equivalent cyber-security issues. Experiments could be done using the DAE with discriminating features like the tonnage of the vessels, its function or its distance from the shore.

## 9 CRediT author statement

**Antoine Chevrot:** Conceptualization, Methodology, Software, Data Curation, Visualization, Writing - Original Draft. **Alexandre Vernotte:** Investigation, Writing - Review & Editing. **Bruno Legeard:** Writing - Review & Editing, Supervision.

## 10 Acknowledgment

This work is part of an ongoing research initiative toward the detection of FDIA in air traffic surveillance communication flows. This work is partially supported by a UBFC-ISITE-BFC SARCoS Grant, the EIPHI Graduate school (contract *ANR-17-EURE-0002*), and ANR GeLeAD project funding. All Computations have been performed on the supercomputer facilities of the Mésocentre de calcul de Franche-Comté.

## 11 Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

- E. W. G. 51. Safety, performance and interoperability requirements document for ads-b/nra application. Technical report, The European Organisation for Civil Aviation Equipment, 2005. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.6059&rep=rep1&type=pdf>.
- S. Akerman, E. Habler, and A. Shabtai. Vizadsb: Analyzing sequences of ads-b images using explainable convolutional lstm encoder-decoder to detect cyber attacks, 2019.
- J. Baek, E. Hableel, Y.-J. Byon, D. S. Wong, K. Jang, and H. Yeo. How to protect ads-b: Confidentiality framework and efficient realization based on staged identity-based encryption. *IEEE Transactions on Intelligent Transportation Systems*, 18(3):690–700, 2017. doi: 10.1109/TITS.2016.2586301.
- E. Cook. Ads-b, friend or foe: Ads-b message authentication for nextgen aircraft. In *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on CyberSpace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, pages 1256–1261, Aug 2015. doi: 10.1109/HPCC-CSS-ICISS.2015.201.
- A. Cretin, A. Vernotte, A. Chevrot, F. Peureux, and B. Legeard. Test data generation for false data injection attack testing in air traffic surveillance. In *4th International Workshop on Testing Extra-Functional Properties and Quality Characteristics of Software Systems (ITEQS 2020)*, Porto, Portugal, mar 2020.
- G. Dan and H. Sandberg. Stealth attacks and protection schemes for state estimators in power systems. In *Smart Grid Communications (Smart-GridComm), 2010 First IEEE International Conference on*, pages 214–219. IEEE, 2010.
- A. Fried and M. Last. Facing airborne attacks on ads-b data with autoencoders. *Computers & Security*, page 102405, 2021. ISSN 0167-4048. doi: <https://doi.org/10.1016/j.cose.2021.102405>. URL <https://www.sciencedirect.com/science/article/pii/S0167404821002297>.
- S. Fute, W. Buhong, Y. Fuhu, and L. Tengyao. Multidevice false data injection attack models of ADS-b multilateration systems. *Security and Communication Networks*, 2019:1–11, mar 2019. doi: 10.1155/2019/8936784.
- E. Habler and A. Shabtai. Using lstm encoder-decoder algorithm for detecting anomalous ADS-B messages. *Computers & Security*, 78:155–173, 2018.
- R. Karam, M. Salomon, and R. Couturier. A comparative study of deep learning architectures for detection of anomalous ads-b messages\*. In *2020 7th International Conference on Control*,



- Decision and Information Technologies (CoDIT)*, volume 1, pages 241–246, 2020. doi: 10.1109/CoDIT49905.2020.9263880.
- M. Kastelic and J. Pers. Building visual anomaly dataset from satellite data using ads-b. In C. Pöpper and M. Strohmeier, editors, *Proceedings of the 7th OpenSky Workshop 2019*, volume 67 of *EPiC Series in Computing*, pages 44–50. EasyChair, 2019. doi: 10.29007/cznf. URL <https://easychair.org/publications/paper/gFzv>.
- M. Leonardi. Ads-b anomalies and intrusions detection by sensor clocks tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 55(5):2370–2381, 2019. doi: 10.1109/TAES.2018.2886616.
- T. Li, B. Wang, F. Shang, J. Tian, and K. Cao. Ads-b data attack detection based on generative adversarial networks. In J. Vaidya, X. Zhang, and J. Li, editors, *Cyberspace Safety and Security*, pages 323–336, Cham, 2019. Springer International Publishing. ISBN 978-3-030-37337-5.
- T. Li, B. Wang, F. Shang, J. Tian, and K. Cao. Dynamic temporal ADS-B data attack detection based on shdp-hmm. *Comput. Secur.*, 93:101789, 2020. doi: 10.1016/j.cose.2020.101789.
- C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou. Autoencoder for words. *Neurocomputing*, 139:84–96, 2014. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2013.09.055>. URL <https://www.sciencedirect.com/science/article/pii/S0925231214003658>.
- F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *2008 eighth IEEE international conference on data mining*, pages 413–422. IEEE, 2008.
- Y. Liu, P. Ning, and M. K. Reiter. False data injection attacks against state estimation in electric power grids. *ACM Transactions on Information and System Security (TISSEC)*, 14(1):13, 2011.
- P. Luo, B. Wang, T. Li, and J. Tian. Ads-b anomaly data detection model based on vae-svdd. *Computers & Security*, 104:102213, 2021. ISSN 0167-4048. doi: <https://doi.org/10.1016/j.cose.2021.102213>. URL <https://www.sciencedirect.com/science/article/pii/S0167404821000377>.
- M. Ma. Resilience against false data injection attack in wireless sensor networks. In *Handbook of Research on Wireless Security*, pages 628–635. IGI Global, 2008.
- P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148*, 2016.
- M. R. Manesh and N. Kaabouch. Analysis of vulnerabilities, attacks, countermeasures and overall risk of the automatic dependent surveillance-broadcast (ADS-B) system. *International Journal of Critical Infrastructure Protection*, 19:16 – 31, 2017. ISSN 1874-5482. doi: <https://doi.org/10.1016/j.ijcip.2017.10.002>. URL <http://www.sciencedirect.com/science/article/pii/S1874548217300446>.
- G. Miebs, M. Mochol-Grzelak, A. Karaszewski, and R. A. Bachorz. Efficient strategies of static features incorporation into the recurrent neural network. *Neural Processing Letters*, 51(3):2301–2316, Jun 2020. ISSN 1573-773X. doi: 10.1007/s11063-020-10195-x. URL <https://doi.org/10.1007/s11063-020-10195-x>.
- M. Monteiro, A. Barreto, R. Division, T. Kacem, J. Carvalho, D. Wijesekera, and P. Costa. Detecting malicious ads-b broadcasts using wide area multilateration. In *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, pages 4A3–1–4A3–12, Sep. 2015. doi: 10.1109/DASC.2015.7311413.
- X. Olive. traffic, a toolbox for processing and analysing air traffic data. *Journal of Open Source Software*, 4:1518, 2019. ISSN 2475-9066. doi: 10.21105/joss.01518.
- X. Olive and L. Basora. Identifying Anomalies in past en-route Trajectories with Clustering and Anomaly Detection Methods. In *ATM Seminar 2019*, VIENNE, Austria, June 2019. URL <https://hal.archives-ouvertes.fr/hal-02345597>.
- X. Olive, J. Grignard, T. Dubot, and J. Saint-Lot. Detecting Controllers’ Actions in Past Mode S Data by Autoencoder-Based Anomaly Detection. In *SESAR Innovation Days 2018*, SALZBURG, Austria, Dec. 2018. URL <https://hal.archives-ouvertes.fr/hal-02338690>.
- D. Park, Y. Hoshi, and C. C. Kemp. A multi-modal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, July 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2801475.
- T. Polishchuk, A. Lemetti, and R. Saez. Evaluation of flight efficiency for stockholm arlanda airport using opensky network data. In C. Pöpper and M. Strohmeier, editors, *Proceedings of the 7th OpenSky Workshop 2019*, volume 67 of *EPiC Series in Computing*, pages 13–24. EasyChair, 2019. doi: 10.29007/9g31. URL <https://easychair.org/publications/paper/hk2Q>.

- M. Schäfer, V. Lenders, and I. Martinovic. Experimental analysis of attacks on next generation air traffic communication. In *International Conference on Applied Cryptography and Network Security*, pages 253–271. Springer, 2013.
- M. Schäfer, M. Strohmeier, V. Lenders, I. Martinovic, and M. Wilhelm. Bringing up opensky: A large-scale ads-b sensor network for research. In *Proceedings of the 13th international symposium on Information processing in sensor networks*, pages 83–94. IEEE Press, 2014.
- M. Schäfer, P. Leu, V. Lenders, and J. Schmitt. Secure motion verification using the doppler effect. In *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, pages 135–145, 2016.
- M. I. Skolnik. Radar handbook, 3rd edition. 2008.
- M. Strohmeier. *Security in Next Generation Air Traffic Communication Networks*. PhD thesis, Oxford University, 2016.
- M. Strohmeier, V. Lenders, and I. Martinovic. Intrusion detection for airborne communication using phy-layer information. In *DIMVA*, 2015a.
- M. Strohmeier, V. Lenders, and I. Martinovic. On the security of the automatic dependent surveillance-broadcast protocol. 17:1066–1087, 2015b. ISSN 2373-745X. doi: 10.1109/COMST.2014.2365951.
- M. Strohmeier, M. Schäfer, R. Pinheiro, V. Lenders, and I. Martinovic. On perception and reality in wireless air traffic communications security. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1338–1357, 2017. doi: 10.1109/TITS.2016.2612584.
- Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19*, page 2828–2837, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330672. URL <https://doi.org/10.1145/3292500.3330672>.
- J. Sun, J. Ellerbroek, and J. Hoekstra. Flight extraction and phase identification for large automatic dependent surveillance–broadcast datasets. *Journal of Aerospace Information Systems*, pages 1–6, 2017.
- A. Vernotte, A. Cretin, B. Legeard, and F. Peureux. A domain-specific language to design false data injection tests for air traffic control systems. *International Journal on Software Tools for Technology Transfer*, Feb 2021. ISSN 1433-2787. doi: 10.1007/s10009-021-00604-4. URL <https://doi.org/10.1007/s10009-021-00604-4>.
- J. Wang, D. Shi, Y. Li, J. Chen, H. Ding, and X. Duan. Distributed framework for detecting pmu data manipulation attacks with deep autoencoders. 10:4401–4410, 2018. ISSN 1949-3061. doi: 10.1109/TSG.2018.2859339.
- K. D. Wesson, T. E. Humphreys, and B. L. Evans. Can cryptography secure next generation air traffic surveillance? *IEEE Security and Privacy Magazine*, 2014.
- L. Xie, Y. Mo, and B. Sinopoli. False data injection attacks in electricity markets. In *Smart Grid Communications (SmartGridComm), First International Conference on*, pages 226–231. IEEE, 2010.
- K. Yang, M. Bi, Y. Liu, and Y. Zhang. Lstm-based deep learning model for civil aircraft position and attitude prediction approach. In *2019 Chinese Control Conference (CCC)*, pages 8689–8694, July 2019. doi: 10.23919/ChiCC.2019.8865874.
- X. Ying, J. Mazer, G. Bernieri, M. Conti, L. Bushnell, and R. Poovendran. Detecting ads-b spoofing attacks using deep neural networks.
- D. Yook, S.-G. Leem, K. Lee, and I.-C. Yoo. Many-to-many voice conversion using cycle-consistent variational autoencoder with multiple decoders. In *Proc. Odyssey 2020 The Speaker and Language Recognition Workshop*, pages 215–221, 2020.
- R. Zhang, G. Liu, J. Liu, and J. P. Nees. Analysis of message attacks in aviation datalink communication. *IEEE Access*, 2017.
- X. Zhang, X. Chen, L. Zhang, and J. Wang. A number-based inventory of particle emissions by civil aviation and the influences on the particle number concentration near zurich airport. In C. Pöpper and M. Strohmeier, editors, *Proceedings of the 7th OpenSky Workshop 2019*, volume 67 of *EPiC Series in Computing*, pages 109–116. EasyChair, 2019. doi: 10.29007/fqt3. URL <https://easychair.org/publications/paper/2fjJ>.
- D. Zhao, J. Sun, and G. Gui. En-route multilateration system based on ads-b and tdoa/aoa for flight surveillance systems. In *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, pages 1–6, 2020. doi: 10.1109/VTC2020-Spring48590.2020.9129436.