Computers & Security 116 (2022) 102652

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

CAE: Contextual auto-encoder for multivariate time-series anomaly detection in air transportation

Antoine Chevrot*, Alexandre Vernotte, Bruno Legeard

DISC/FEMTO-ST Institute, UBFC, CNRS Besançon, France

ARTICLE INFO

Article history:
Received 28 July 2021
Revised 23 December 2021
Accepted 8 February 2022
Available online 11 February 2022

Keywords:
Anomaly detection
Multivariate time series
ADS-B
CAE
Air traffic security

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 Contextual Auto-Encoder (CAE). 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 CAE's efficiency, an evaluation dataset was created using real-life anomalies as well as realistically crafted trajectory modifications, with which the CAE as well as three anomaly detection models from the literature were evaluated. Results show that the CAE 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.

1. Introduction

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.

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 in 2002[1] 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 cybersecurity was not in the highest priority at the time. 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.

* Corresponding author.
E-mail addresses: Antoine.Chevrot@femto-st.fr (A. Chevrot), Alexandre.Vernotte@femto-st.fr (A. Vernotte), Bruno.Legeard@femto-st.fr (B. Legeard).

[1] https://www.icao.int/publications/Documents/9750_2ed_en.pdf.

https://doi.org/10.1016/j.cose.2022.102652
0167-4048/© 2022 Elsevier Ltd. All rights reserved.

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., 0000).

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 CAE* –Contextual Auto-Encoder–, a novel asymmetric auto-encoder addressing contextual 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, with a focus on trajectory modifications. 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 CAE 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 ML 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 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 1090 MHz 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 80 NM longitudinal separation to just 5 NM in NRAs) than what is presently used with current procedures (Procedural Separation) (EUROCAE, 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 cybersecurity issues. Extensive research can be found in the literature that discuss these issues (Schäfer et al., 2013; Strohmeier et al., 2017; Zhang 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 blackouts 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 (Manesh and Kaabouch, 2017; Strohmeier, 2016). 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 interfer-

Table 1
Different attack scenarios from the taxonomy of Schäfer et al. (2013).

Attack Scenario	Method	Severity	Complexity	References
Ghost Aircraft Injection	Message Injection	High	Low	Costin and Francillon (2012); Schäfer et al. (2013); McCallie et al. (2011)
Ghost Aircraft Flooding	Message Injection	High	Low	Costin and Francillon (2012); Schäfer et al. (2013); McCallie et al. (2011)
Virtual Trajectory Modification	Message Modification	High	Medium	Schäfer et al. (2013); Wilhelm et al. (2012); Pöpper et al. (2011)
False Alarm Attack	Message Modification	Medium	Medium	Costin and Francillon (2012); Schäfer et al. (2013); Wilhelm et al. (2012)
Aircraft Disappearance	Message Deletion	High	Low	Schäfer et al. (2013); McCallie et al. (2011)
Aircraft Spoofing	Message Modification	High	Low	Costin and Francillon (2012); Schäfer et al. (2013); Purton et al. (2010)

ence. 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.

The above three techniques allow the execution of several attack scenarios (Schäfer et al., 2013) that can be categorized in the following taxonomy (cf. Table 1):

- **Ghost Aircraft Injection.** Creation of a non-existing aircraft by broadcasting fake ADS-B messages on the communication channel.
- **Ghost Aircraft Flooding.** This attack is similar to the first one but consists of injecting multiple aircraft simultaneously with the goal of saturating the air situation picture and thus generates a denial of service of the controller's surveillance system.
- **Virtual Trajectory Modification.** Using either message modification or a combination of message injection and deletion, the objective is to modify the trajectory of an aircraft, for instance to simulate an emergency scenario.
- **False Alarm Attack.** Modification of the messages of an aircraft in order to indicate a fake alarm. A typical example would be modifying the squawk code to 7500, indicating the aircraft has been hijacked.
- **Aircraft Disappearance.** Deletion of all messages emitted by an aircraft can lead to the failure of collision avoidance systems and ground sensors confusion. It could also force the aircraft under attack to land for safety check.
- **Aircraft Spoofing.** Spoofing of the ICAO number of an aircraft through message deletion and injection. This could allow an enemy aircraft to pass for a friendly one and reduce causes for alarm when picked up by PSR.

One can sense the potential for disaster if one of these operations were 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 communication flow while being able to maintain the infrastructure in a working state.

This work attempts to detect anomalies in ADS-B data that could be resulting from Ghost Aircraft Injection or Virtual trajectory modifications, as described in the taxonomy above. These FDIAs can be realized in real life using two main methods: overshadowing or bit-flipping. Overshadowing is done technically by an attacker sending a signal stronger than the original one, resulting a part or all of the targeted messages to be replaced. Bit-flipping on the other hand uses the targeted signal and superimposes it to itself converting bits from 1 to 0 or 0 to 1. Both result into arbitrary data being injected without the knowledge of the participants, aircraft or ATC. Wilhelm et al. (2012) demonstrated the feasibility of those attacks, hence stressing the necessity to have systems detecting them. In this work, a simulator is used to create data from a FDIA.

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 ML 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 ML 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 framework 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

Compared to other kind of solutions, ML techniques are most of the time data-driven and applied directly to the data carried by the ADS-B protocol, also called logical layer, trying to find abnormalities in the time-series received by air traffic controllers, which could be due to FDIAs.

The subject of anomaly detection using statistical or ML techniques is a subject than can be found in many application domains and taking many forms. In 2009, Chandola et al. (2009) published a survey presenting several different research on anomaly detection, both generic and applied to specific fields. In this survey, they separate anomalies in different categories, each with specific properties. The difference between those categories are mainly based on their nature: individual outliers among a set of data for instance are called point anomalies, while data being abnormal depending on a specific context are called contextual anomalies. A third type of anomalies called collective anomalies is a group of data considered abnormal while individuals inside this group might be considered normal. In the case of anomaly detection in ADS-B data, the first type is often due to encoding errors and can be considered as outlier data to clean-up before model training. Contextual anomalies are usually the main anomaly type concerning time-series, including the aim of this paper: the virtual trajectory modifications. The last type of anomaly is also relevant to be detected as some message modifications could fall under this category.

Many traditional anomaly detection methods are not fit to detect these anomalies in time-series. For instance, most of distance-based or clustering methods, by themselves, like DBSCAN (Ester et al., 1996), or IMS (Iverson et al., 2012) are mostly effective to detect individual outliers in the data but will disregard contextual anomalies. Additionally, they are usually suffering from the curse of dimensionality, which is troublesome when working on big time-series dataset. Similarly, ensemble methods like Isolation Forests (Liu et al., 2012), though they can be used when coupled with sliding windows (Ding and Fei, 2013) to manage time-dependencies, are also falling behind other techniques to properly detect contextual abnormalities.

To detect contextual anomalies in time-series, one approach would be to transform them first into point anomalies and then use one of the previous techniques cited. For instance, in cell phone fraud detection, Fawcett and Provost (1996) use cell phone usage records to create rules which are use to contextualize data. Once contextualized, the data are then processed by a linear threshold unit to determine if they are fraudulent or not. Similarly, Ma and Perkins (2003) transform time-series into a feature space which can then be processed by a one-class SVM to detect anomalies.

Another way to tackle contextual anomalies is to utilize the structure of the time-series itself. To do so, during its training, a model is only given normal data, with regular pattern given a context. Once trained, this model is expected to behave improperly when given abnormal data, hence raising an alert. For instance, statistical models such as ARIMA (Bianco et al., 2001) can be trained to predict future values of a given sequence. ARIMA computes predictions based on a given test sequence and compares it against the real value to raise an alarm or not. Zhao et al. (2017) uses a similar statistical approach to make short-term state forecasting on power micro-grids to detect FDIAs. Other efforts like this one can actually be found to predict FDIAs in smart-grids whether using residual functions coupled with a threshold (Ameli et al., 2018) or using deep neural networks (Yu et al., 2018).

Reconstruction-based methods work in a similar fashion. As described by Pimentel et al. (2014), reconstruction methods use a lower dimension representation of the original data which helps model to separate normal representation from anomalous ones. They are separated in two different categories: subspace-based models and neural networks. While subspace models like PCA can be used to detect anomalies (Dutta et al., 2007), it is most of the time used to reduce the dimensions of a given problem to then use other methods like clustering (Jarry et al., 2020). On the other hand, neural networks and more precisely auto-encoders are widely applied methods to detect anomalies, including anomalies in time-series. Auto-encoders (AE) are neural network models that have the same number of input and output neurons, with smaller hidden layer that will compress the original data. The assumption being that anomalies are troublesome to reconstruct, it is then easy to detect them by comparing input and output of AE models. One can already find efforts to detect FDIAs using auto-encoders in different domains (Habler and Shabtai, 2018; Kundu et al., 2020)

As stated by Janakiraman and Nielsen (2016), the challenges of detecting anomalies in ADS-B messages is mainly due to the high-dimensionality, the time-dependencies and the multivariate nature of the data. Most of the traditional methods of detecting anomalies do not tackle properly all these challenges except for the more recent neural-network / deep-learning methods (Chalapathy and Chawla, 2019). Same conclusion in their survey presenting anomaly solutions in aviation applications, Basora et al. (2019) place a great emphasis on auto-encoders to help securing ADS-B protocol.

Often coupled with Recurrent Neural Network (RNN) like LSTM or GRU (Malhotra et al., 2016), auto-encoders have shown good accuracy to detect coarse anomalies (Habler and Shabtai, 2018), or more specific behaviours (Olive and Basora, 2019; Olive et al., 2018). 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 these presented auto-encoder 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 contextual feature. This architecture stems from Yook et al. (2020) work on separating the sound received by speakers placed differently and 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 in a given context, addressing the variability

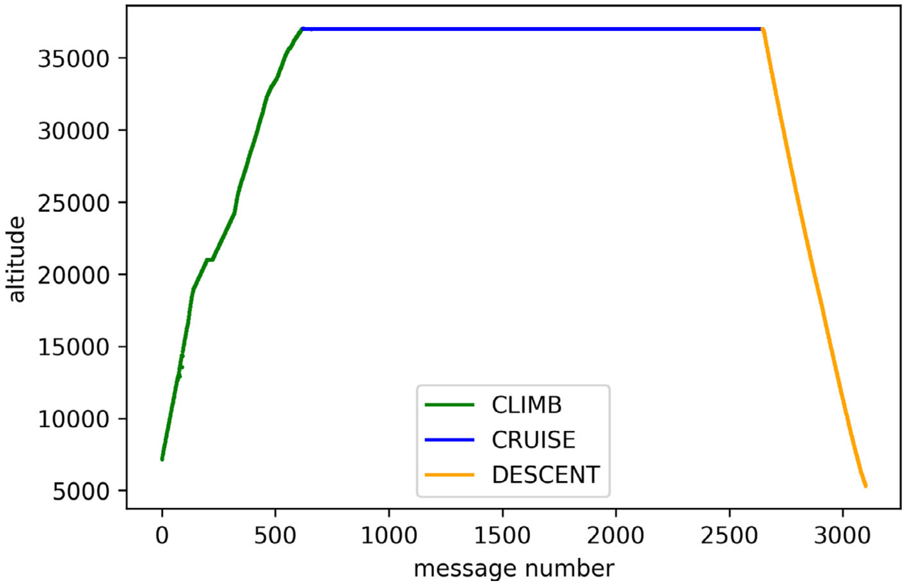

Fig. 1. The 3 different phases of a flight used as contextual feature.

of the time series over certain period of time, resulting into better detection.

4. CAE: contextual auto-encoder

To detect FDIAs in ADS-B time-series, the context in which the data are issued is primordial. Indeed, a sudden important drop in altitude is perfectly normal in the context of a descending phase, while it is quite abnormal in the context of an ascending or cruising phase. From the related work discussed previously, while neural network models like auto-encoders are well suited to detect FDIAs, they are usually trained with as many training examples as possible, disregarding the context of the input data.

In this section, the Contextual Auto-Encoder (CAE) is introduced as a new mean to take advantage of the benefits of the auto-encoder architecture while taking into account the context of the input data by using several decoders instead of a single one.

4.1. Problem statement

As stated previously, 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 $\mathbf{x} = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}\}$, where an observation $\mathbf{x_t} \in \mathbf{x}$ is an M-dimensional vector at time $t (t \leq N)$, *i.e.* $\mathbf{x_t} = [x_t^1, x_t^2, \ldots, x_t^M]$ 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 $\{\mathbf{x_{t-T}}, \mathbf{x_{t-T+1}}, \ldots, \mathbf{x_t}\}$ 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, Fig. 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 contextual feature $\mathbf{C_{t-T:t}}$ 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 contextual feature in this paper.

4.2. Model architecture

The basis of the CAE 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 contextual feature presented previously. This section explains the different parts of the model that can be found in the Fig. 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 contextual data is a one-dimensional array which associate each time-window to a contextual feature.

② **Sorting Layer:** The sorting layer separates the batches of windows into mini-batches according to the contextual 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 contextual feature can take. For instance, the contextual 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 s 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

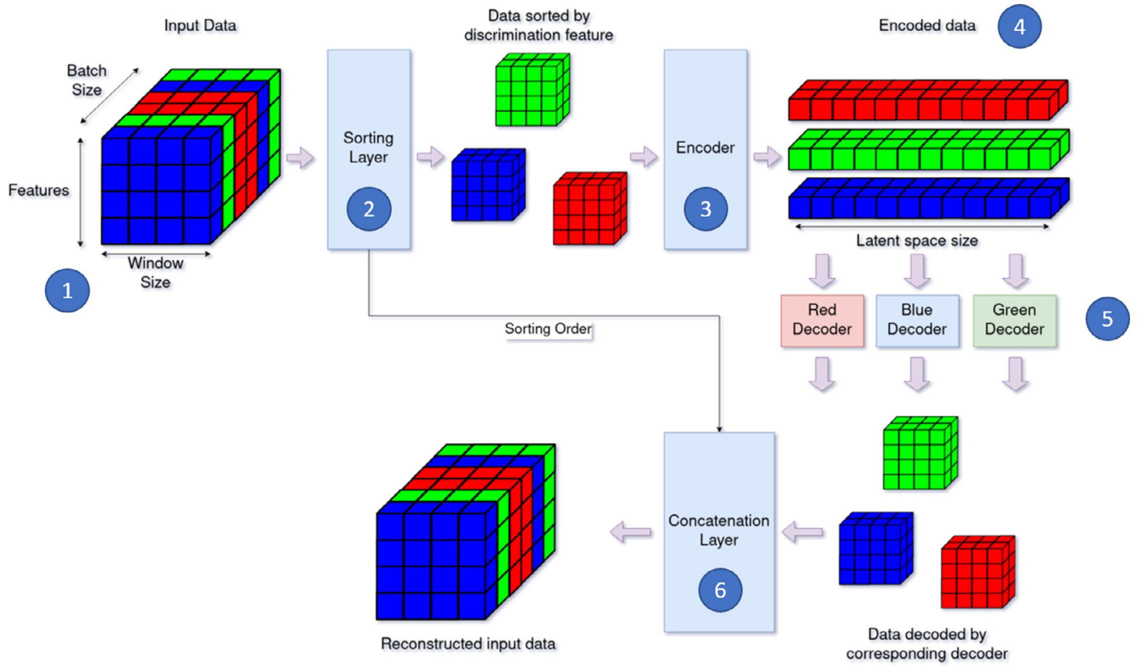

Fig. 2. Architecture of the CAE model.

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 CAE 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 CAE, 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 **Contextual Decoders:** The decoding part of the CAE is mainly what makes it different from a regular 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 CAE 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. Input and output of the CAE model

The CAE model has all the characteristics of a classical auto-encoder and as such can be used for a wide-range of use-cases as long as the data include some kind of contextual feature to take advantage of having several decoders. Except from that, the input data do not have any restriction and are solely at the discretion of the user. For instance, to detect anomalies in the ADS-B protocol, the input features are flight data like altitude, speed etc. They are exhaustively described in Section 5.4. They are encoded and then decoded by the different decoders to then be processed by the anomaly detection part. The hyperparameters for the training are also specified in Section 6.1 for replicability.

4.4. 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:

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

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

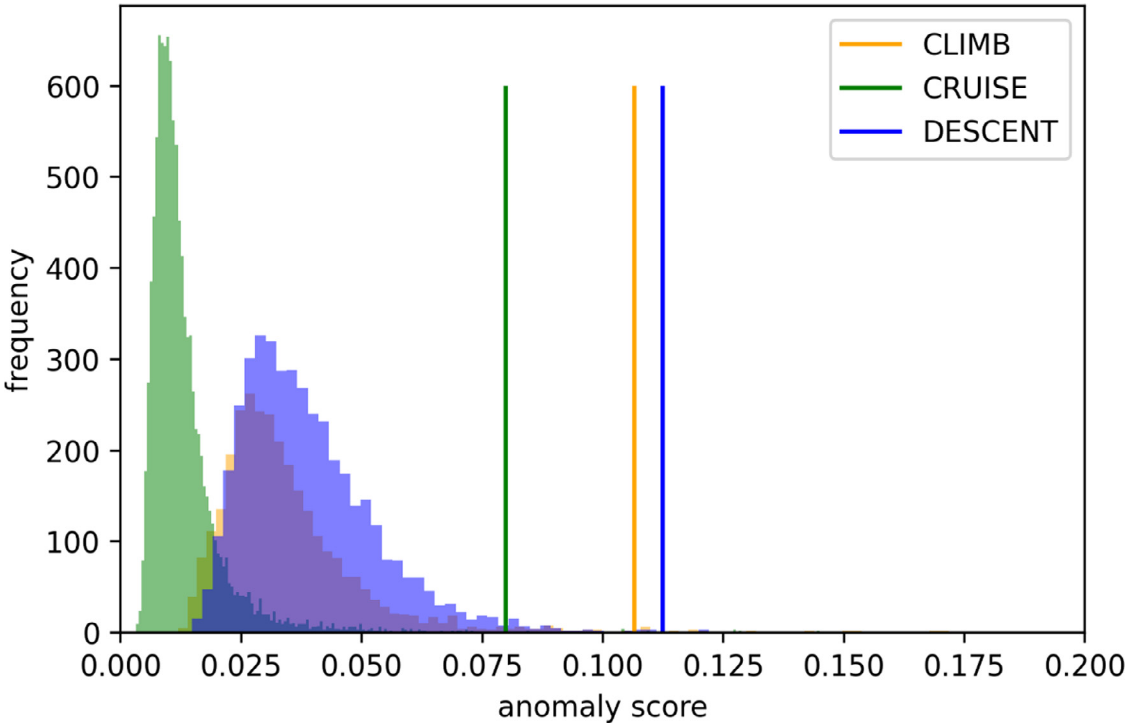

Fig. 3. The anomaly score's distributions of the training data for each phase along with its calculated thresholds.

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 scarse, 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 CAE. 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 τ is defined as $\tau = \mu + 3\sigma$ where μ is the mean of the anomaly score distribution of one decoder and σ 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 Fig. 3), assures the 3-sigma rule to yield a low false positive rate while being sensitive enough to flag anomalies.

5. Dataset creation and pre-processing

Apart from the CAE, 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.

5.1. Global architecture

In Fig. 4 is described the overall architecture used to train the CAE. After retrieving data from the Opensky Network, the data preprocessing 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

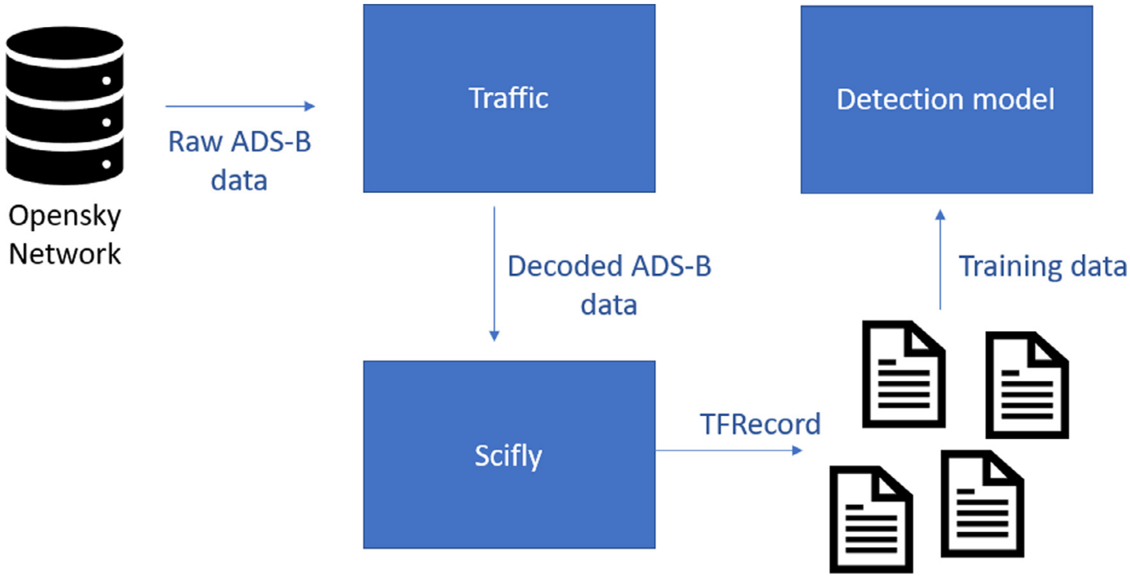

Fig. 4. The data architecture used for gathering and processing ADS-B messages into training data.

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 s of recording. Changing the original rate of messages to 2 s 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[2] 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[3] to serialize data making it available to a large share of ML 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 (Vernotte et al., 2021) jointly with Smartesting (https://www.smartesting.com) and Kereval (https://www.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 ML 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 2. 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

[2] https://github.com/Wirden/scifly.

[3] https://developers.google.com/protocol-buffers/.

Table 2
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.

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)	Istambul (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

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 contextual 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 km away from the departure or arrival airport. This helps when the fuzzy logic labels a crash as a simple descending maneuver. Fig. 1 shows the different phases of a flight automatically determined by the Scifly 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 two different subset: an FDIA set and a baseline set. The FDIA set is composed of data made from the FDI-T module that simulates data that would be resulting from a FDIA including different Trajectory modification scenario an attacker could create. All the anomalies contained in this set are applied on all the flights found in the training set, but from January 2021. On the other hand, the baseline includes real-life data, both normal to evaluate the models' accuracy on regular data and abnormal to verify the presence of tool bias in the crafted FDIAs.

① **FDIAs**

Gradual Drift (DRIFT) – Attack that consists of simulating an altitude drift or a velocity drift. The altitude or velocity messages on the attacked time window are all raised/lower 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 Fig. 6 shows a velocity drift used during the evaluation

Made-up Crashes (CRASH) – Using FDI-T, life-like crashes scenarios can be faked 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. Fig. 7 shows some of the features modified during a CRASH attack.

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

② **Baseline**

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 –.

Ryanair Hijack (HJK) – Constituted of the Ryanair flight 4978 from Athens to Vilnius which was forcibly diverted to Minsk after entering the Belarus airspace on the 23rd of May 2021.[4] It is to be noted that the emergency was turned on by the crew 2 min 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. This is not an FDIA as it is actually a real-life abnormal scenario but detecting this anomaly is impor-

[4] https://www.flightradar24.com/blog/ryanair-flight-4978-to-vilnius-forcibly-diverted-to-minsk/.

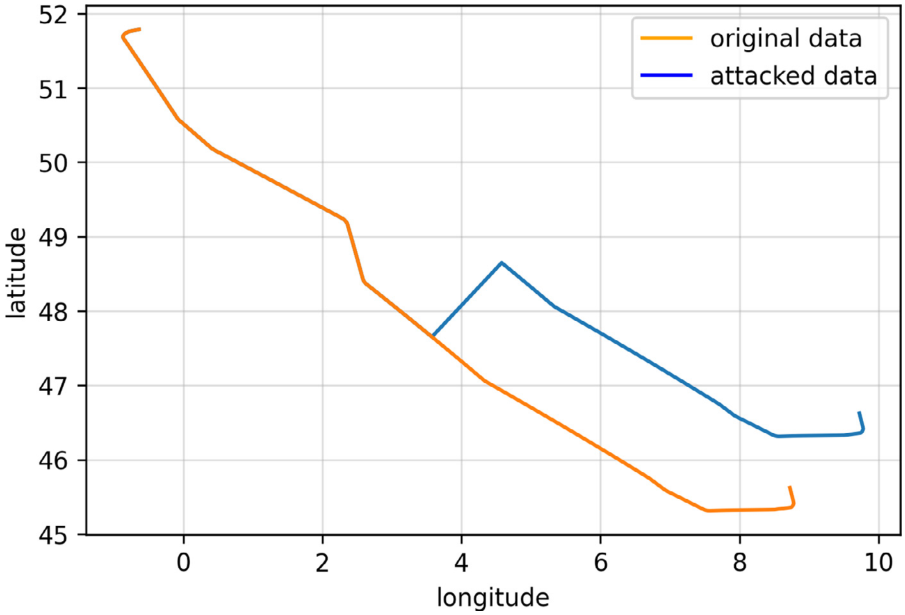

Fig. 5. Constant position offset attack.

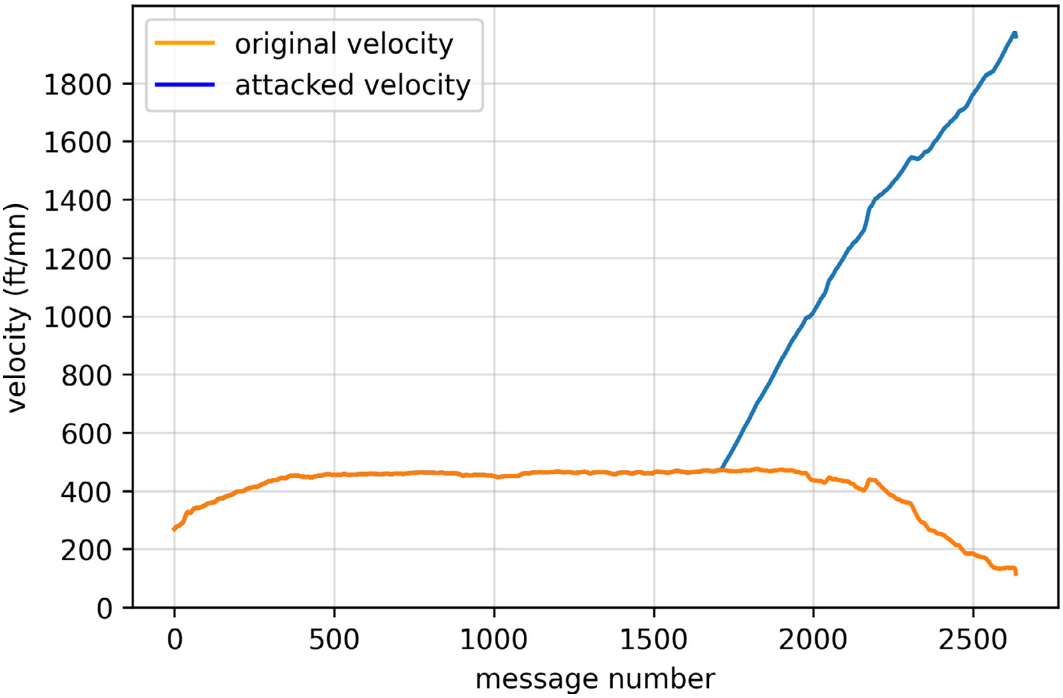

Fig. 6. Velocity drift attack.

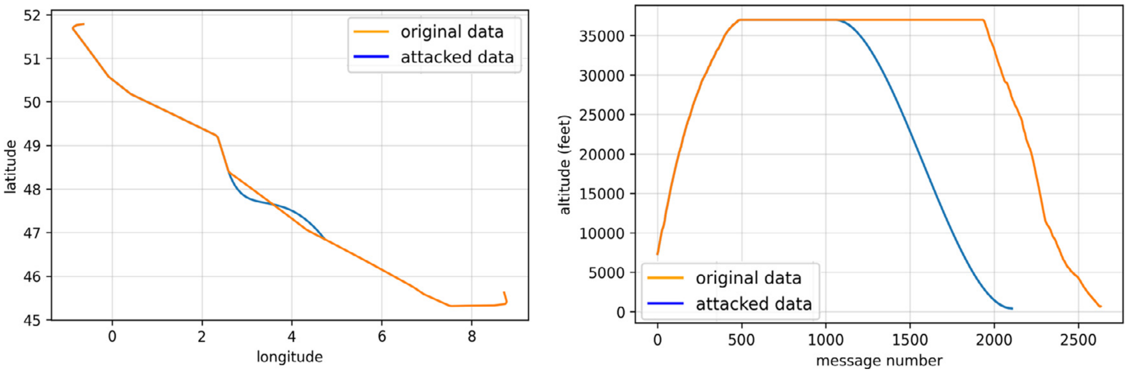

Fig. 7. Crash attack. Latitude / longitude on the left and the altitude on the right. Other features like vertical speed or track are also modified *realistically*.

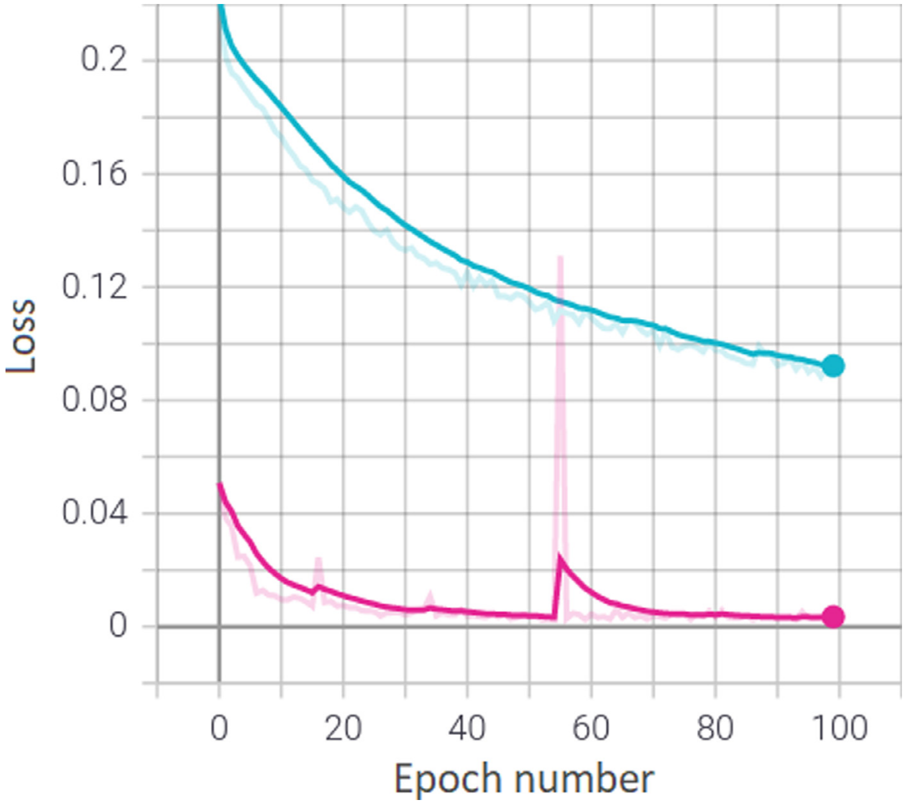

Fig. 8. The training loss per epoch. In blue (top) is the training loss, in pink (bottom) the validation loss. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

tant to prove that the model is not biased into detecting only anomalies created by FDI-T.

6. Experimental evaluation

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

6.1. Model training specifications

The CAE model was trained with the training dataset abovementioned which represent 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 7.8 TeraFLOPS. The training on average was taking around 26 min per epoch. In Fig. 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 much higher than its validation counterpart.

To train the CAE 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:

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

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

All the results and implementation of this paper are accessible on the Scifly[5] 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 baseline dataset for future models in the growing field of anomaly detection for false data injected in ADS-B data.

6.2. Baseline models results

To show the overall performance of the CAE, it is compared with 3 other unsupervised approaches for anomaly detection for false data injected 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 3 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

[5] https://github.com/Wirden/scifly.

Table 3
Comparison of the different models evaluated.

Evaluation Dataset	Evaluation metric	LSTM-AE	IForest	VAE-SVDD	CAE
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	**0.778**	0.729	0.152	**0.439**
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	**0.937**	0.926	**0.939**
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	**0.598**	0.107	**0.097**
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	**0.925**	0.573	**0.955**
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	**0.797**	0.440	**0.738**

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 CAE compared with the state-of-the-art approaches in FDIA detection in ADS-B. 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 below). 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 an encoder-decoder reconstruction used by Habler and Shabtai (2018) to detect false data 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 helps raising the F1 score but would result to a FPR being too high for anomaly detection.

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 trivial 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. It is however not well equipped to tackle the time dependency of the data. 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 CAE'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.

6.3. CAE results against the baseline

Compared to the LSTM-AE with a single decoder, the CAE, thanks to its specialized decoders, manages to discriminate anomalous situations like crashes from regular descent operations while keeping a very similar low FPR overall. However, from the metrics alone, the CAE seems to be under-performing compared to the LSTM-AE for the hijack scenario. This can be explained by looking at Fig. 9 which compares the anomaly score over the message windows for both models. One can observe that for the CAE, the anomaly is set off before the actual emergency due to its delay with the diversion of the flight. It explains the FPR being much higher than the other models and displays the reactivity of the CAE 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 the CAE does not label the end of the flight as abnormal from its ADS-B data.

Compared to the VAE-SVDD, the CAE performs better on all dataset except on the constant position offset where all models perform poorly due to the scenario's very nature. Indeed, the small offset added to the latitude and longitude is not enough to trigger alarms leading to extremely low F1 scores. This attack can only

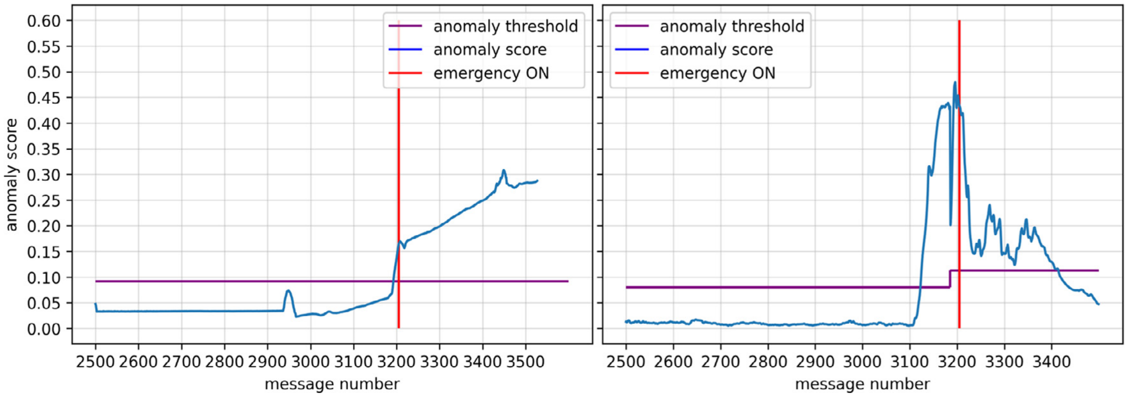

Fig. 9. Anomaly score for the LSTM-AE on the left and the CAE on the right for the Ryanair hijack flight.

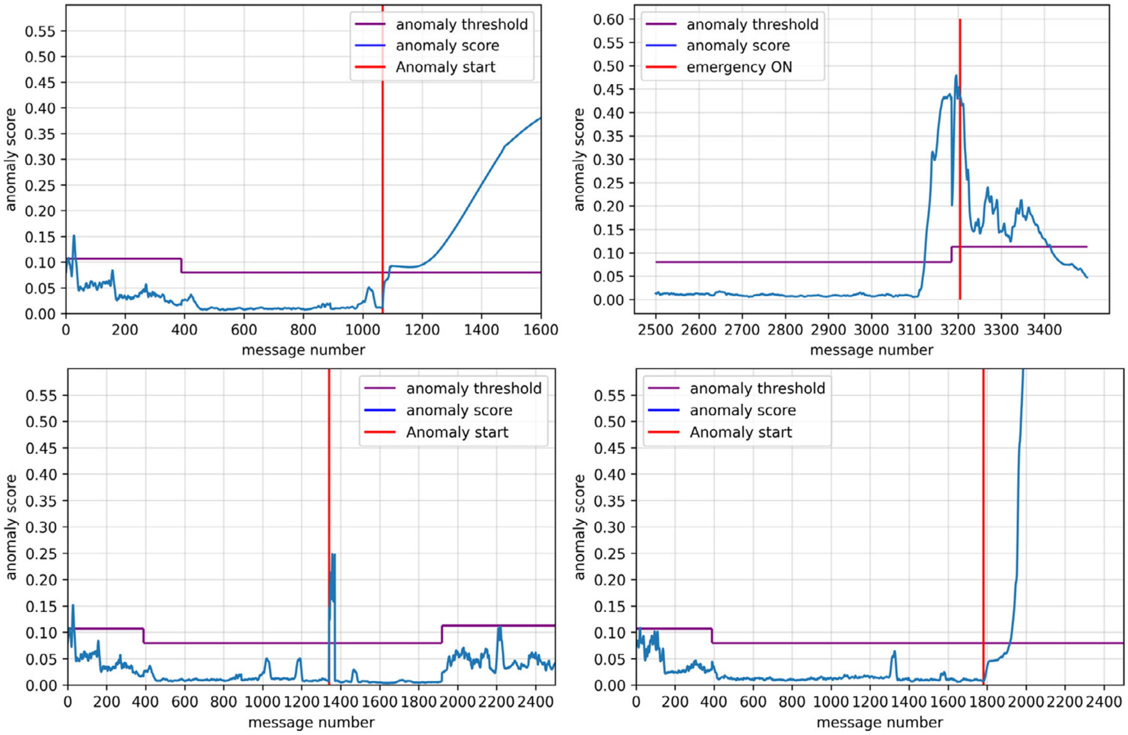

Fig. 10. CAE anomaly scores for a flight taken randomly from the different evaluation dataset. The top-left figure is from the CRASH dataset, the top-right is from the Ryanair hijack, the bottom-left is from the constant position offset and the bottom-right is from the velocity drift dataset

be detected by the LSTM based models when the values actually changes.

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 CAE, limiting its usage in real-life applications. In conclusion, while the CAE does not well perform on the position offset FDIA, it has the best accuracy on normal data and the best scores on other FDIAs. It is also the only model that manages to detect the real-life anomalous scenario as soon as it started, leading to an alarm raised before the change of flight status. This shows that the CAE is a conclusive model to detect crafted attacks in ADS-B messages.

7. Discussion

The CAE 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 CAE 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 CAE 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 CAE 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 attacks without ever have false positives. On Fig. 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 attack 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, which would otherwise go undetected.

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 CAE, 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 ML models implementation, also fully open-accessible. The results presented on this dataset show that the CAE 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 contextual 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 CAE. Weather data could be added to better isolate storm avoidance manoeuvres from anomalous situations. The use of the COMM-B 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 CAE 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 cybersecurity issues. Experiments could be done using the CAE with contextual features like the tonnage of the vessels, its function or its distance from the shore.

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.

CRediT authorship contribution statement

Antoine Chevrot: Conceptualization, Methodology, Software, Data curation, Visualization, Writing – original draft. **Alexandre Vernotte:** Investigation, Writing – review & editing. **Bruno Legeard:** Writing – review & editing, Supervision.

Acknowledgements

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 (ANR-18-ASTR-0011) project funding. All Computations have been performed on the supercomputer facilities of the Mésocentre de calcul de Franche-Comté.

References

Akerman, S., Habler, E., Shabtai, A., 2019. VizADS-B: analyzing sequences of ADS-B images using explainable convolutional LSTM encoder-decoder to detect cyber attacks. arXiv:1906.07921

Ameli, A., Hooshyar, A., El-Saadany, E.F., Youssef, A.M., 2018. Attack detection and identification for automatic generation control systems. IEEE Trans. Power Syst. 33 (5), 4760–4774. doi:10.1109/TPWRS.2018.2810161.

Baek, J., Hableel, E., Byon, Y.-J., Wong, D.S., Jang, K., Yeo, H., 2017. How to protect ADS-B: confidentiality framework and efficient realization based on staged identity-based encryption. IEEE Trans. Intell. Transp. Syst. 18 (3), 690–700. doi:10.1109/TITS.2016.2586301.

Basora, L., Olive, X., Dubot, T., 2019. Recent advances in anomaly detection methods applied to aviation. Aerospace 6 (11). doi:10.3390/aerospace6110117. https://www.mdpi.com/2226-4310/6/11/117

Bianco, A., Garcia Ben, M., Martinez, E.J., Yohai, V., 2001. Outlier detection in regression models with ARIMAerrors using robust estimates. J. Forecast. 20. doi:10.1002/for.768.

Chalapathy, R., Chawla, S., 2019. Deep learning for anomaly detection: a survey. arXiv:1901.03407

Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: a survey. ACM Comput. Surv. 41 (3). doi:10.1145/1541880.1541882.

Cook, E., 2015. 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, pp. 1256–1261. doi:10.1109/HPCC-CSS-ICESS.2015.201.

Costin, A., Francillon, A., 2012. Ghost in the Air (Traffic): On Insecurity of ADS-BProtocol and Practical Attacks on ADS-B Devices. Black Hat USA, pp. 1–12.

Cretin, A., Vernotte, A., Chevrot, A., Peureux, F., Legeard, B., 2020. Test data generation for false data injection attack testing in air traffic surveillance. 4th International Workshop on Testing Extra-Functional Properties and Quality Characteristics of Software Systems (ITEQS 2020), Porto, Portugal.

Dan, G., Sandberg, H., 2010. Stealth attacks and protection schemes for state estimators in power systems. In: Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on. IEEE, pp. 214–219.

Ding, Z., Fei, M., 2013. An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. IFAC Proc. Vol. 46 (20), 12–17. doi:10.3182/20130902-3-CN-3020.00044. 3rd IFAC Conference on Intelligent Control and Automation Science ICONS 2013 https://www.sciencedirect.com/science/article/pii/S1474667016314999

Dutta, H., Giannella, C., Borne, K., Kargupta, H., 2007. Distributed Top-K outlier detection from astronomy catalogs using the DEMAC system. 10.1137/1.9781611972771.47

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. AAAI Press, pp. 226–231.

EUROCAE, 2005. Safety, Performance and Interoperability requirements Document for ADS-B/NRA Application. Technical Report. The European Organisation for Civil Aviation Equipment. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.6059&rep=rep1&type=pdf

Fawcett, T., Provost, F.J., 1996. Combining data mining and machine learning for effective user profiling. KDD.

Fried, A., Last, M., 2021. Facing airborne attacks on ADS-B data with autoencoders. Comput. Secur. 102405. doi:10.1016/j.cose.2021.102405. https://www.sciencedirect.com/science/article/pii/S0167404821002297

Fute, S., Buhong, W., Fuhu, Y., Tengyao, L., 2019. Multidevice false data injection attack models of ADS-B multilateration systems. Secur. Commun. Netw. 2019, 1–11. doi:10.1155/2019/8936784.

Habler, E., Shabtai, A., 2018. Using LSTM encoder-decoder algorithm for detecting anomalous ADS-B messages. Comput. Secur. 78, 155–173.

Iverson, D., Martin, R., Schwabacher, M., Spirkovska, L., Taylor, W., Mackey, R., Castle, J., 2012. General purpose data-driven system monitoring for space operations. J. Aerosp. Comput., Inf., Commun. 9. doi:10.2514/1.54964.

Janakiraman, V.M., Nielsen, D., 2016. Anomaly detection in aviation data using extreme learning machines. In: 2016 International Joint Conference on Neural Networks (IJCNN), pp. 1993–2000. doi:10.1109/IJCNN.2016.7727444.

Jarry, G., Delahaye, D., Nicol, F., Feron, E., 2020. Aircraft atypical approach detection using functional principal component analysis. J. Air Transp. Manag. 84, 101787. doi:10.1016/j.jairtraman.2020.101787. https://www.sciencedirect.com/science/article/pii/S0969699719303266

Kundu, A., Sahu, A., Serpedin, E., Davis, K., 2020. A3d: Attention-based auto-encoder anomaly detector for false data injection attacks. Electr. Power Syst. Res. 189, 106795. doi:10.1016/j.epsr.2020.106795. https://www.sciencedirect.com/science/article/pii/S0378779620305988

Leonardi, M., 2019. ADS-B anomalies and intrusions detection by sensor clocks tracking. IEEE Trans. Aerosp. Electron. Syst. 55 (5), 2370–2381. doi:10.1109/TAES.2018.2886616.

Li, T., Wang, B., Shang, F., Tian, J., Cao, K., 2019. ADS-B data attack detection based on generative adversarial networks. In: Vaidya, J., Zhang, X., Li, J. (Eds.), Cyberspace Safety and Security. Springer International Publishing, Cham, pp. 323–336.

Liou, C.-Y., Cheng, W.-C., Liou, J.-W., Liou, D.-R., 2014. Autoencoder for words. Neurocomputing 139, 84–96. doi:10.1016/j.neucom.2013.09.055. https://www.sciencedirect.com/science/article/pii/S0925231214003658

Liu, F.T., Ting, K.M., Zhou, Z.-H., 2008. Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining. IEEE, pp. 413–422.

Liu, F.T., Ting, K.M., Zhou, Z.-H., 2012. Isolation-based anomaly detection. ACM Trans. Knowl. Discov. Data 6 (1). doi:10.1145/2133360.2133363.

Liu, Y., Ning, P., Reiter, M.K., 2011. False data injection attacks against state estimation in electric power grids. ACM Trans. Inf. Syst. Secur. (TISSEC) 14 (1), 13.

Luo, P., Wang, B., Li, T., Tian, J., 2021. ADS-B anomaly data detection model based on VAE-SVDD. Comput. Secur. 104, 102213. doi:10.1016/j.cose.2021.102213. https://www.sciencedirect.com/science/article/pii/S0167404821000377

Ma, J., Perkins, S., 2003. Time-series novelty detection using one-class support vector machines. In: Proceedings of the International Joint Conference on Neural Networks, vol. 3, pp. 1741–1745 vol.3. doi:10.1109/IJCNN.2003.1223670.

Ma, M., 2008. Resilience against false data injection attack in wireless sensor networks. In: Handbook of Research on Wireless Security. IGI Global, pp. 628–635.

Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., Shroff, G., 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. arXiv preprint arXiv:1607.00148

Manesh, M.R., Kaabouch, N., 2017. Analysis of vulnerabilities, attacks, countermeasures and overall risk of the automatic dependent surveillance-broadcast (ADS-B) system. Int. J. Crit. Infrastruct. Prot. 19, 16–31. doi:10.1016/j.ijcip.2017.10.002. http://www.sciencedirect.com/science/article/pii/S1874548217300446

McCallie, D., Butts, J., Mills, R., 2011. Security analysis of the ADS-B implementation in the next generation air transportation system. Int. J. Crit. Infrastruct. Prot. 4 (2), 78–87.

Miebs, G., Mochol-Grzelak, M., Karaszewski, A., Bachorz, R.A., 2020. Efficient strategies of static features incorporation into the recurrent neural network. Neural Process. Lett. 51 (3), 2301–2316. doi:10.1007/s11063-020-10195-x.

Monteiro, M., Barreto, A., Division, R., Kacem, T., Carvalho, J., Wijesekera, D., Costa, P., 2015. Detecting malicious ADS-B broadcasts using wide area multilateration. In: 2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC) doi:10.1109/DASC.2015.7311413.

Olive, X., 2019. Traffic, a toolbox for processing and analysing air traffic data. J. Open Source Softw. 4, 1518. doi:10.21105/joss.01518.

Olive, X., Basora, L., 2019. Identifying anomalies in past en-route trajectories with clustering and anomaly detection methods. In: ATM Seminar 2019. VIENNE, Austria. https://hal.archives-ouvertes.fr/hal-02345597

Olive, X., Grignard, J., Dubot, T., Saint-Lot, J., 2018. Detecting controllers' actions in past mode S data by autoencoder-based anomaly detection. SESAR Innovation Days 2018. SALZBURG, Austria. https://hal.archives-ouvertes.fr/hal-02338690

Park, D., Hoshi, Y., Kemp, C.C., 2018. A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. IEEE Robot. Autom. Lett. 3 (3), 1544–1551. doi:10.1109/LRA.2018.2801475.

Pimentel, M.A., Clifton, D.A., Clifton, L., Tarassenko, L., 2014. A review of novelty detection. Signal Process. 99, 215–249. doi:10.1016/j.sigpro.2013.12.026. https://www.sciencedirect.com/science/article/pii/S016516841300515X

Pöpper, C., Tippenhauer, N.O., Danev, B., Capkun, S., 2011. Investigation of signal and message manipulations on the wireless channel. In: Atluri, V., Diaz, C. (Eds.), Computer Security – ESORICS 2011. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 40–59.

Purton, L., Abbass, H., Alam, S., 2010. Identification of ADS-B system vulnerabilities and threats. ATRF 2010: 33rd Australasian Transport Research Forum.

Schäfer, M., Lenders, V., Martinovic, I., 2013. Experimental analysis of attacks on next generation air traffic communication. In: International Conference on Applied Cryptography and Network Security. Springer, pp. 253–271.

Schäfer, M., Leu, P., Lenders, V., Schmitt, J., 2016. Secure motion verification using the doppler effect. In: Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks, pp. 135–145.

Schäfer, M., Strohmeier, M., Lenders, V., Martinovic, I., Wilhelm, M., 2014. 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. IEEE Press, pp. 83–94.

Strohmeier, M., 2016. Security in Next Generation Air Traffic Communication Networks. Oxford University Ph.D. thesis.

Strohmeier, M., Lenders, V., Martinovic, I., 2015. Intrusion detection for airborne communication using PHY-layer information. In: DIMVA.

Strohmeier, M., Lenders, V., Martinovic, I., 2015. On the security of the automatic dependent surveillance-broadcast protocol 17, 1066–1087. doi:10.1109/COMST.2014.2365951.

Strohmeier, M., Schäfer, M., Pinheiro, R., Lenders, V., Martinovic, I., 2017. On perception and reality in wireless air traffic communications security. IEEE Trans. Intell. Transp. Syst. 18 (6), 1338–1357. doi:10.1109/TITS.2016.2612584. arXiv:1602.08777v3.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D., 2019. 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. Association for Computing Machinery, New York, NY, USA, pp. 2828–2837. doi:10.1145/3292500.3330672.

Sun, J., Ellerbroek, J., Hoekstra, J., 2017. Flight extraction and phase identification for large automatic dependent surveillance–broadcast datasets. J. Aerosp. Inf. Syst. 14 (10), 566–572.

Vernotte, A., Cretin, A., Legeard, B., Peureux, F., 2021. A domain-specific language to design false data injection tests for air traffic control systems. Int. J. Softw. Tools Technol. Trans. doi:10.1007/s10009-021-00604-4.

Wang, J., Shi, D., Li, Y., Chen, J., Ding, H., Duan, X., 2018. Distributed framework for detecting PMU data manipulation attacks with deep autoencoders 10, 4401–4410. doi:10.1109/TSG.2018.2859339.

Wilhelm, M., Schmitt, J.B., Lenders, V., 2012. Practical message manipulation attacks in IEEE802.15.4 wireless networks. In: In Proceedings of MMB & DFT 2012.

Xie, L., Mo, Y., Sinopoli, B., 2010. False data injection attacks in electricity markets. In: Smart Grid Communications (SmartGridComm), First International Conference on. IEEE, pp. 226–231.

Yang, K., Bi, M., Liu, Y., Zhang, Y., 2019. LSTM-based deep learning model for civil aircraft position and attitude prediction approach. In: 2019 Chinese Control Conference (CCC), pp. 8689–8694. doi:10.23919/ChiCC.2019.8865874.

Skolnik, M. I., 2008. Radar Handbook, third ed.. ↾McGraw-Hill Professional. ISBN-10: ↾9780071485470.

Ying, X., Mazer, J., Bernieri, G., Conti, M., Bushnell, L., Poovendran, R.,. Detecting ADS-B spoofing attacks using deep neural networks. arXiv:1904.09969v1.

Yook, D., Leem, S.-G., Lee, K., Yoo, I.-C., 2020. Many-to-many voice conversion using cycle-consistent variational autoencoder with multiple decoders. In: Proc. Odyssey 2020 The Speaker and Language Recognition Workshop, pp. 215–221.

Yu, J.J.Q., Hou, Y., Li, V.O.K., 2018. Online false data injection attack detection with wavelet transform and deep neural networks. IEEE Trans. Ind. Inf. 14 (7), 3271–3280. doi:10.1109/TII.2018.2825243.

Zhang, R., Liu, G., Liu, J., Nees, J.P., 2017. Analysis of message attacks in aviation datalink communication. IEEE Access 6, 455–463. doi:10.1109/ACCESS.2017.2767059.

Zhao, D., Sun, J., Gui, G., 2020. En-route multilateration system based on ADS-B and TDOA/AOA for flight surveillance systems. In: 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), pp. 1–6. doi:10.1109/VTC2020-Spring48590.2020.9129436.

Zhao, J., Zhang, G., La Scala, M., Dong, Z.Y., Chen, C., Wang, J., 2017. Short-term state forecasting-aided method for detection of smart grid general false data injection attacks. IEEE Trans. Smart Grid 8 (4), 1580–1590. doi:10.1109/TSG.2015.2492827.

Antoine Chevrot, received his Engineer's degree in computer science from the Institut National des Sciences Appliquées (INSA) in Lyon, France in 2016 and is currently pursing a Ph.D. in artificial intelligence applied to the air traffic security in the University of Franche-Comté, Besançon, France. His research are mostly focused on ADS-B protocol and how to detect anomalies using auto-encoder based architectures.

Alexandre Vernotte, obtained his Ph.D. in 2015 in the field of security testing for web application at the university of Franche-Comté. He then continued his cybersecurity related works on complex systems for 2 years as a postodoctoral fellow at the Royale Institute of Technology (KTH) in Stockholm, Sweden. Since 2018, he works in a postdoctoral position at the University of Franche-Comté, first on testing and detecting techniques against False Data Injection Attacks (FDIA) and then on the study of artificial intelligence applied to the prioritisation of tests.

Bruno Legeard, is Professor at Université Bourgogne Franche-Comté - Institut FEMTO-ST (France), and Scientific Advisor at Smartesting. Bruno has more than 20 years' expertise in Model-Based Testing/Model-Based Security Testing (MBT/MBST) and its introduction in the industry. His research activities mainly concern features about automation of Model-Based Test case generation using AI techniques. His research results in more than 100 scientific and industrial publications based on MBT, MBST and AI-driven Testing.