2024 Integrated Communications, Navigation and Surveillance Conference (ICNS) | 979-8-3503-9309-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICNS60906.2024.10550816

Detecting Injection Attacks in ADS-B Devices Using RNN-Based Models

Tala Talaei Khoei[1], Hadjar Ould Slimane [2], Khair Al Shamaileh [3], Vijaya Kumar Devabhaktuni[4], and Naima Kaabouch[2]

[1]Khoury College of Computer Sciences, Roux Institute at Northeastern University, Portland, ME 04101 USA
[2]School of Electrical Engineering and Computer Science, University of North Dakota, Grand Forks 58202, ND, USA
[3] Electrical and Computer Engineering Department, Illinois State University, Normal, IL 61761 USA
[4] Electrical and Computer Engineering Department, Illinois State University, Normal, IL 61761 USA

***Abstract*— The Automatic Dependent Surveillance Broadcast (ADS-B) system is a critical communication and surveillance technology used in the Next Generation (NextGen) project as it improves the accuracy and efficiency of air navigation. These systems allow air traffic controllers to have more precise and real-time information on the location and movement of aircraft, leading to increased safety and improved efficiency in the airspace. While ADS-B has been made mandatory for all aircraft in the Federal Aviation Administration (FAA) monitored airspace, its lack of security measures leaves it vulnerable to cybersecurity threats. Particularly, ADS-B signals are susceptible to false data injection attacks due to the lack of authentication and integrity measures, which poses a serious threat to the safety of the National Airspace System (NAS). Many studies have attempted to address these vulnerabilities; however, machine learning and deep learning approaches have gained significant interest due to their ability to enhance security without modifying the existing infrastructure. This paper investigates the use of Recurrent Neural Networks for detecting injection attacks in ADS-B data, leveraging the time-dependent nature of the data. The paper reviews previous studies that used different machine learning and deep learning techniques and presents the potential benefits of using RNN algorithms to improve ADS-B security.**

Keywords*— *Automatic dependent surveillance broadcast, gated recurrent unit, injection attacks, long short-term memory, recurrent neural networks, time series.

I. INTRODUCTION

Automatic Dependent Surveillance Broadcast (ADS-B) is a revolutionary technology designed to improve communication and surveillance within the aviation industry. Developed as a crucial component of the Next Generation (NextGen) project, its primary goal is to enhance the efficiency and reliability of air navigation [1]. Additionally, ADS-B aims to reduce the maintenance costs associated with air traffic control infrastructure [2]. Recognizing its immense potential for advancing aviation safety, the Federal Aviation Administration (FAA) and the European Aviation Safety Agency (EASA) have made it mandatory for all aircraft operating within their monitored airspace to be equipped with ADS-B devices by 2020 [3] [4]. This mandate ensures widespread adoption of the technology, enabling a more comprehensive and streamlined approach to air traffic management.

By leveraging ADS-B, aircraft can autonomously broadcast vital information, such as their Global Positioning System (GPS)-derived positions, airspeed, altitude, and identification data. Ground-based receivers and other equipped aircraft can then access and utilize this real-time data, resulting in enhanced situational awareness for pilots and air traffic controllers alike. The seamless sharing of crucial flight information facilitates more precise and efficient routing, ultimately contributing to safer and more reliable air travel. ADS-B represents a significant leap forward in modernizing the aviation industry, and its implementation is a critical step toward achieving a more interconnected and technologically advanced air transportation system.

Prior to the implementation of ADS-B, traditional radar systems only updated aircraft positions once every few seconds, resulting in less accurate and potentially outdated information for pilots. With ADS-B, pilots receive more accurate and frequent position updates at a frequency of at least one message per second [5], allowing them to make better-informed decisions and avoid potential hazards [1]. While ADS-B has proven to be a significant advancement in air navigation, its design did not prioritize security, leaving it susceptible to various cybersecurity threats.

One of the key vulnerabilities lies in the fact that data transmitted through the 1090ES datalink is unencrypted, making it accessible to all parties on the network [1]. This lack of encryption exposes ADS-B data to potential risks, including unauthorized manipulation, deletion, or injection of information [6-7]. Furthermore, the absence of authentication techniques during data transmission and reception makes the ADS-B system vulnerable to unauthorized access and potential security breaches. The lack of robust authentication opens the door for malicious actors to interfere with the data exchange process and possibly compromise the integrity and reliability of the entire system.

Thus, addressing these security concerns is crucial to ensuring the safe and dependable operation of ADS-B. Given the vulnerabilities of ADS-B and the need to safeguard airspace from potential malicious attacks, finding efficient solutions is of utmost importance. Over the past years, numerous studies have been presented in the literature to address this issue, offering various approaches, such as traffic modeling [8], group validation [9], physical layer fingerprinting [10], and data fusing [11].

However, many of these methods require modifications to the existing ADS-B infrastructure, prior knowledge of the system, or the addition of extra hardware [6]. with the increasing availability of publicly accessible ADS-B data, machine learning, and deep learning approaches have emerged as promising alternatives for enhancing security without necessitating any changes to the current ADS-B infrastructure. Leveraging the power of these advanced computational techniques, these approaches can analyze and interpret vast amounts of ADS-B data in real time, identifying patterns and anomalies that could indicate potential security threats.

By harnessing machine learning and deep learning, it becomes possible to proactively detect and respond to suspicious activities or unauthorized intrusions within the ADS-B network. These algorithms can continuously learn from historical data, adapt to evolving threats, and improve their accuracy over time, bolstering the security of the entire airspace system. For example, in [12], the authors explored various machine learning models to detect jamming on ADS-B systems, including logistic regression, artificial neural networks, support vector machine, k-nearest neighbor, and decision trees. The results indicated that a two-hidden-layer neural network with 15 neurons outperformed all other techniques.

In [13], a Bi-directional Long-Short Time Memory (Bi-LSTM) model was proposed to detect track outliers in ADS-B data, along with a multidimensional outlier descriptor based on the dynamic time warping algorithm. This method achieved acceptable results. In [14], a deep learning-based approach was introduced to identify three types of ADS-B spoofing attacks: message replay attacks, ghost aircraft injection attacks, and aircraft spoofing attacks. The model, consisting of a two-layer neural network provides better performance, compared to the other techniques.

Even though the proposed methods in detecting and classifying false data injection attacks on ADS-B provided high performance, there are a limited number of studies that mainly focus on time series data. It is worth mentioning that ADS-B data is time-dependent; therefore, proposing any Deep Learning (DL) models, relying on time series data can be important in detecting and classifying injection attacks on ADS-B. Motivated by the dependency of ADS-B on times series data, this study widely proposes four DL models, namely Long Short-Term Memory (LSTM), Bi-directional LSTM (Bi-LSTM), Gated Recurrent Unit (GRU), and Bi-directional GRU (Bi-GRU). In short, the main key contributions of this study are as follows:

- Proposing four DL models, depending on time series data, namely LSTM, Bi-LSTM, GRU, and Bi-GRU,
- Providing a comprehensive comparison of these models in terms of accuracy, probability of detection, misdetection, false alarm, training time, testing time, and memory during training and testing.

The remainder of the paper is structured as follows. The methods used in this work, such as data collecting, data preprocessing, modeling, and performance evaluation of RNN models, are described in Section II. While the research results and their discussion are presented in section III. Lastly, section III offers a thorough conclusion.

II. METHODOLOGY

This section provides an overview of the process of collecting ADS-B data and the preprocessing steps involved in making the data suitable for RNN models. The ADS-B data is collected using receivers placed at various locations, and it contains information about the position, altitude, velocity, and other parameters of the aircraft. The first preprocessing step involves data cleaning, which removes any noisy or irrelevant data points that may negatively impact the performance of the RNN models.

The second preprocessing step is standardization, which scales the data to a common range, between zero and one, to make it easier for the RNN models to process. Then, it discusses the different RNN algorithms used for this purpose, Gated Recurrent Unit (GRU) Long Short-Term Memory (LSTM), Bi-Long Short-Term Memory (Bi-LSTM), and Bi-Gated Recurrent Unit (Bi-GRU).

II.1.1 ADS-B Data Collection and Pre-processing Techniques

ADS-B data was collected and preprocessed as previously described in [20]. The corresponding dataset consists of 22,315 samples with s with equally

distributed two classes (11,158 authentic messages and 11,157 Injection attacks). In the given data, there are three types of injection attacks, namely path modification, ghost aircraft injection, and velocity drift. This dataset underwent several data pre-processing techniques, including data cleaning, feature extraction, standardization, and encoding, as highlighted in [20]. Additionally, a feature selection process was also conducted to optimize the accuracy and computational complexity of the RNN algorithms used.

II.1.2 Modeling and Performance Evaluation of RNN Models

The focus of this study is a class of DL models, namely Recurrent Neural Network (RNN), which was developed to handle sequential data. They are designed to process sequences by maintaining an internal state that can capture information from past inputs and pass it along to future inputs [21]. This allows RNN models to maintain context and capture patterns in sequential data. In a traditional neural network, all inputs and outputs are assumed to be independent of one another.

However, RNN models are recurrent since they carry out the same calculations for each element of a sequence, where the outcome is dependent on past calculations, which makes these models have a memory that can retain information about previous calculations [21]. Another benefit of RNN models is that they can handle inputs of varying lengths, making them ideal for processing sequential data. They have the ability to generalize the information learned from one sequence to another, even if the sequences are of different lengths.

RNN training utilizes the backpropagation over time to update the weights of the model, which is a similar algorithm to the backpropagation algorithm that is used in traditional neural networks. In this algorithm, the parameters are shared by all timesteps; therefore, the gradient at each output is affected not only by the current time step but also by prior ones. As a result, RNN models struggle to learn long-term dependencies due to the vanishing or expanding gradient issue, which occurs when the gradients become very small or big during backpropagation, making it difficult for the network to learn [22]. To overcome this problem, newer architectures that employ gating methods to govern the flow of information between neural network cells are utilized, like GRU and LSTM models [22].

II.1.3 2.2.1. LSTM

LSTM is a type of RNN model that was proposed to overcome the problem of vanishing or exploding gradients in traditional RNNs caused by the capturing of both pertinent and irrelevant information [23]. LSTM introduces memory cells and gates that can hold their state across several time steps and govern the flow of information into and out of the memory cells [24]. LSTMs have memory cells that may be regarded as a form of internal state that the model can utilize to recall information [24].

LSTMs also feature input, input modulation, forget, and output gates that regulate information flow into and out of the memory cell [25]. These input and output gates allow the model to select what information from the current time step is fed to the memory cell, by taking the input, combining it with the previous hidden state, and passing it through a sigmoid function. The input modulation gate determines how much of the new input should be added to the cell state. It is similar to the input gate, but it uses a tanh activation function [25].

On the other hand, the forget gate determines which information from the previous time step should be removed from the memory cell. It takes the previous hidden state and the current input and passes it through a sigmoid function [25]. The output gate determines what information should be utilized for the next time step from the memory cell, enabling it to choose whether to keep or forget knowledge from prior time steps, which makes it understand long-term dependencies in sequential data that was a challenging task in the past [26]. As a result, LSTMs are an extremely effective tool for modeling complicated, non-linear dependencies in sequential data.

II.1.4 Bi-LSTM model

One of the other known time series models is Bi-LSTM. This model can learn from the sequence of data in both the backward and forward directions [23]. In Bi-LSTM, the given data flows in two directions, resulting in differentiation from the simple LSTM. In the simple LSTM, the given data only flows in a designated direction (forward or backward).

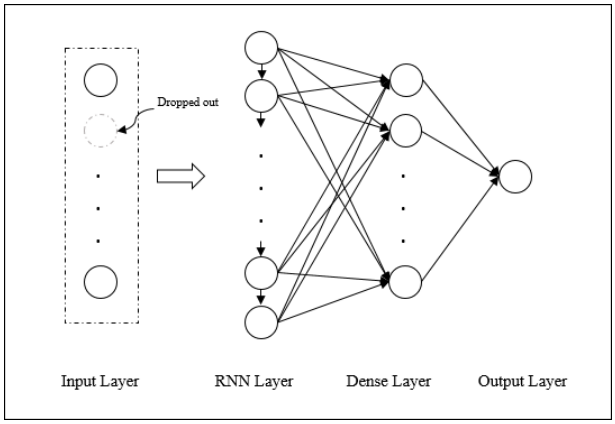

Fig. 1. General Structure of the Models.

In contrast, Bi-LSTM can take the given data flows from both directions to preserve past and future data. In general, Bi-LSTM uses two-linked layers and predicts the sequence of each element by applying a finite sequence based on the input elements in the past and future [23, 27]. This model can perform as two LSTM models that run in parallel. In other words, one of these models runs from left to right, and the other model runs from right to left. The targeted class prediction, composite output, can use the forward function with the inputs L and H hidden units [23, 27, 28].

II.1.5 GRU Model

The GRU model is another popular type of RNN that is more computationally efficient and has a simpler topology than LSTMs [11]. A GRU cell is comprised of only two gates that lower the gating signals and associated parameters: the reset gate and the update gate [29]. The hidden state output at time t, like the LSTM cell, is computed using the hidden state of time t-1 and the input time series value at time t [25]. To decrease the number of parameters, the GRU cell incorporates the LSTM cell's forget gate and input gate as an update gate. GRU cell is less powerful than the original LSTM since it has fewer gates. Therefore, the GRU can't be taught to count or answer context- free language problems.

II.1.6 Bi-GRU Model

The Bi-GRU model consists of two GRU layers that process the input sequence in two opposite directions, forward and backward [30]. Each GRU layer consists of multiple GRU cells, and each cell takes as input the current input vector and the hidden state from the previous time step. During the forward pass, the input sequence is fed into the forward GRU layer, and the hidden state of each cell is updated based on the input and the previous hidden state. At the same time, the backward GRU layer processes the input sequence in reverse, and the hidden state of each cell is updated based on the input and the previous hidden state. The output of the forward and backward GRU layers are then concatenated to produce the final output sequence. This architecture allows the model to capture dependencies in both directions of the input sequence, which can exploit information both from the past and the future [31].

In this work, a dropout layer was used after the input layer in each model architecture as a means of regularization and to avoid overfitting the training data. The concept of dropout involves the temporary removal of units within a neural network. This means that a unit and all its incoming and outgoing connections are dropped out or excluded from the network. This technique aims to prevent overfitting by forcing the network to learn more robust features, which helps it generalize better to new data [32].

Additionally, the learning rate has been increased to accelerate the training process, and the momentum has been raised to enhance the model's ability to overcome local minima and converge to the global minimum of the loss function. These modifications aim to improve the model's generalization performance and prevent overfitting, resulting in a more robust and accurate system. Furthermore, the addition of a dense layer was added after each RNN layer helped the model learn the internal connections

of the data and boost the accuracy, as shown in Fig. 1. Moreover, random search has been employed to find the optimal hyperparameters in order to obtain the highest accuracy and the minimum loss. The performance of these models is evaluated using the accuracy, probabilities of detection, misdetection, and false alarm, as well as, training time, detection time, memory usage during training, and memory usage during detection [33, 34].

III. Results and Discussion

In this work, the dataset was divided into two subsets - a training set and a testing set. The training set consists of 67% of the data, which was used to train the RNN model, while the remaining 33% of the data was set aside for testing the performance of the trained model. To evaluate the performance of those models, different evaluation metrics were utilized namely, Probability of Detection (PoD), Probability of Misdetection (PoM), Probability of False Alarm (PoFA), and Accuracy (Acc).

The probability of correctly categorizing injected messages divided by the total number of injected messages is defined as PoD. On the other hand, PoM is the proportion of injected messages assessed as genuine over the total amount of injected messages. The percentage of authentic communications that were erroneously categorized over the total number of legitimate messages is given by PoFA. The proportion of successfully categorized messages over the entire number of messages is denoted as Acc.

$$PoD = \frac{TP}{+ FN} \times 100 \quad (1)$$

$$\frac{TP\ FN}{} \times 100 \quad (2)$$

$$PoM = \frac{}{TP + FN}$$

$$PoFA = \frac{FP}{TN + \mathbf{P}} \times 1100 \quad (3)$$

$$Acc = \frac{TP + TN}{TP + FN + FP + \mathbf{N}} \times 100100 \quad (4)$$

Where TP and TN correspond to the number of accurately classified malicious and legitimate messages, respectively, whereas FN and FP indicate the number of erroneously predicted malicious and legitimate messages In this study, the RNN models are constructed using Keras and they are trained on an Intel i7-10700, 2.90GHz CPU. As shown in Table I, those models employ stochastic gradient descent as an optimizer with a learning rate of 0.05, and a momentum of 0.8. The activation function used to produce an output value is sigmoid and the number of epochs used to update the network for the entire training dataset is 180; whereas the batch size is 200.

TABLE I. Parameters Setting

Parameter	Setting
Epochs	180
Batch size	200
Learning-rate	0.05
Momentum	0.8
Optimizer	Stochastic gradient descent

Fig. 2 shows the results of the confusion matrices for Bi-GRU, GRU, Bi-LSTM, and LSTM. Fig. 3 presents the results of the performance evaluation of the selected models in terms of their accuracy, probabilities of detection, misdetection, and false alarm rates. It is noteworthy that all models demonstrated good overall performance with an accuracy rate of over 92% and a probability of detection of over 91%. However, it is observed that the GRU and Bi-LSTM models outperform the others with the highest accuracy rate of 94.61% and the highest probability of detection rate of 95.26%, respectively.

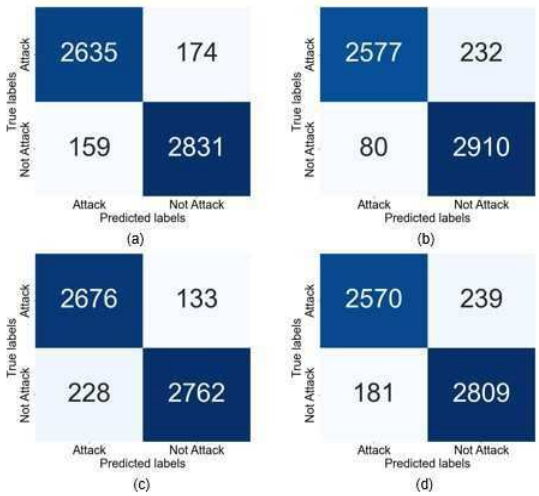

Fig. 2. Confusion matrix of (a) Bi-GRU, (b) GRU, (c) Bi-LSTM and (d) LSTM.

In addition, it was found that the GRU and Bi-LSTM models not only demonstrated the highest accuracy and probabilities of detection but they also achieved the lowest rates of false alarms and misdetection. Specifically, the GRU model had a false alarm rate of 2.70%, while the Bi-LSTM model had a misdetection rate of 4.73%. According to these findings, the GRU and Bi-LSTM models are the most effective at classifying the data points and detecting the attacks. This information is critical for making

decisions in situations where precise detection and classification are paramount.

Table II provides a summary of the results obtained from the performance evaluation of the four RNN classifiers. This table presents the key metrics and measures that were used to assess the performance of the classifiers, including training time, detection time, memory usage during training, and memory usage during detection. It can be observed that the GRU model has the best performance in terms of time and used memory in the training phase, while the Bi-GRU model has the best performance in terms of time and used memory in the prediction phase. The optimized computational performance of the GRU and Bi-GRU models can be attributed to their simplified architecture compared to the LSTM models since they use a simplified gating mechanism that requires fewer parameters, making them computationally efficient.

Based on the findings of the study, the evaluation of the performance of the GRU, LSTM, Bi-GRU, and Bi-LSTM models highlights the trade-offs between computational efficiency and accuracy in deep learning applications. This observation suggests that the choice of the optimal model depends on the specific requirements of the application and which aspects are more important. The faster training and prediction times and lower memory usage of the GRU and Bi-GRU models make it ideal for applications where computational efficiency is of utmost importance and where real-time performance is a priority.

On the other hand, the high accuracy and low misdetection of the Bi- LSTM model make it more suitable for applications where it is crucial to accurately detect attacks. In our case, ensuring high accuracy and low misdetection rates were deemed the most critical performance metrics, since missing an attack could result in catastrophic consequences. In particular, a missed attack could result in mid-air collisions, which could lead to loss of life and significant damage. Given the severity of the potential impact of missed attacks, it is vital to minimize misdetection rates to reduce the risk of such incidents.

VI. Conclusion

In this paper, we proposed a comparative study for detecting false data injection attacks on ADS-B systems based on four RNN models, LSTM, GRU, Bi-LSTM, and Bi-GRU. The RNN models were trained and tested on the previously collected and prepossessed ADS-B data that includes legitimate and malicious samples from three different types of injection attacks, namely, path modification, ghost aircraft injection, and velocity drift. Out of those models, the Bi-LSTM model was able to detect injection attacks with high accuracy and a low misdetection probability. From this study's results, it can be concluded that detecting anomalies in ADS-B signals can be efficiently done without altering the ADS-B protocol or compromising the integrity of the existing infrastructure.

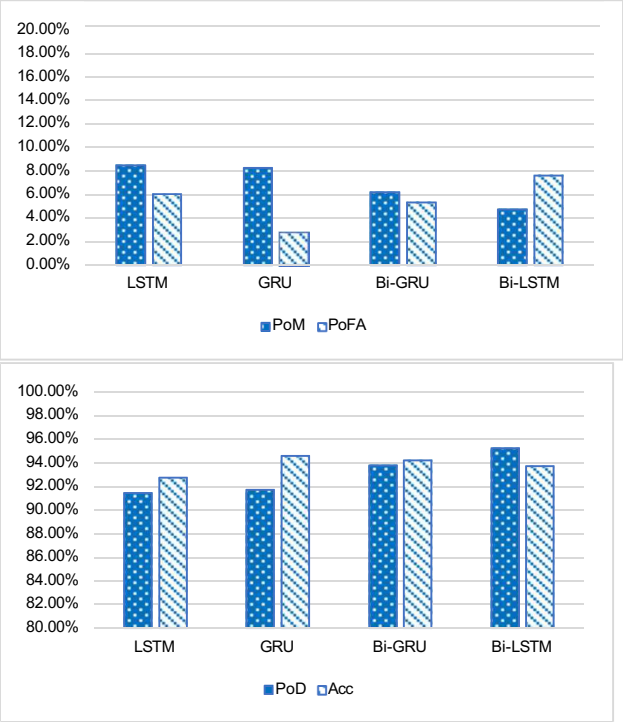

Fig. 2. Results of RNN classifiers in detecting ADS-B attacks.

TABLE II. RNN Models Performance Results

Metrics	LSTM	GRU	Bi-GRU	Bi-LSTM
PoD	91.49%	91.74%	93.80%	**95.26%**
PoM	8.50%	8.25%	6.19%	**4.73%**
PoFA	6.05%	**2.70%**	5.31%	7.62%
Acc	92.75%	**94.61 %**	94.25%	93.77%
T_t **(s)**	86.6	**53.2**	91.7	88.8
T_d **(s)**	0.7	0.7	**0.5**	0.6
Mem_t (MiB)	122.8	**109.2**	159.6	158.4
Mem_d (MiB)	0.148	0.129	**0.070**	0.125

Acknowledgments

The authors acknowledge the support of the National Science Foundation (NSF), Award Number 2006674.

References

[1] Federal Aviation Administration, US Department of Transportation, "Advisory Circular: Automatic Dependent Surveillance-Broadcast Operations," no. 90-114B, 2019.

[2] Z. Wu, T. Shang and A. Guo, "Security Issues in Automatic Dependent Surveillance - Broadcast (ADS-B): A Survey," in IEEE Access, vol. 8, pp. 122147-122167, 2020, doi: 10.1109/ACCESS.2020.3007182.

[3] Federal Aviation Administration, "Automatic Dependent Surveillance-Broadcast ADS-B Out Performance Requirements to Support Air Traffic Control ATC Service," Final Rule, 14 CFR Part 91, Federal Register, vol. 75(103), May 2010.

[4] European Aviation Safety Agency, “Certification Considerations for the Enhanced ATS in Non-Radar Areas using ADS-B Surveillance (ADS-B-NRA) Application via 1,090 MHz Extended Squitter,” AMC 20-24, Germany, 2008.
[5] Code of Federal Regulations, title 14, chapter I, subchapter F, part 91.
[6] M. R. Manesh and N. Kaabouch, “Cyber-attacks on unmanned aerial system networks: Detection, countermeasure, and future research directions,” Computers & Security, vol. 85, pp. 386–401, 2019.
[7] M Strohmeier, M Schafer, V Lenders, and I Martinovic, “Realities and challenges of NextGen air traffic management: The case of ADS-B,” in IEEE Communications Magazine, vol. 52, no. 5, pp. 111–11, 2014
[8] M. Leonardi and G. Sirbu, “ADS-B Crowd-Sensor Network and Two-Step Kalman Filter for GNSS and ADS-B Cyber-Attack Detection,” Sensors, vol. 21, no. 15, p. 4992, Jul. 2021
[9] Y. A. Nijsure, G. Kaddoum, G. Gagnon, F. Gagnon, C. Yuen, R. Mahapatra, “Adaptive air-to-ground secure communication system based on ads-b and wide-area multilateration”, IEEE Transactions on Vehicular Technology 65 (5) (2016) 3150–3165.
[10] M. Strohmeier, V. Lenders and I. Martinovic, "Intrusion Detection for Airborne Communication Using PHY-Layer Information," in Detection of Intrusions and Malware, and Vulnerability Assessment: 12th International Conference Proceedings, vol. 12, pp. 67-77, Springer International Publishing, 2015.
[11] D. Jeon, Y. Eun, and H. Kim, “Estimation fusion with radar and ADS-B for air traffic surveillance,” International Journal of Control, Automation and Systems, vol. 13, no. 2, pp. 336-345, 2015.
[12] M.R. Manesh, M.S. Velashani, E. Ghribi and N. Kaabouch, "Performance Comparison of Machine Learning Algorithms in Detecting Jamming Attacks on ADS-B Devices," in 2019 IEEE International Conference on Electro Information Technology (EIT), pp. 200-206, May 2019.
[13] Y. Cao, J. Cao, Z. Zhou, and Z. Liu, "Aircraft Track Anomaly Detection Based on MOD-Bi-LSTM," Electronics, vol. 10, no. 9, p. 1007, 2021.
[14] X. Ying, J. Mazer, G. Bernieri, M. Conti, L. Bushnell and R. Poovendran, "Detecting ADS-B Spoofing Attacks Using Deep Neural Networks," in 2019 IEEE Conference on Communications and Network Security (CNS), pp. 187-195, June 2019.
[15] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of CNN and RNN for natural language processing," arXiv preprint arXiv:1702.01923, 2017.
[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," arXiv preprint arXiv:1406.1078, 2014.
[17] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N.E.Y. Soplin, R. Yamamoto, X. Wang, and S. Watanabe, "A comparative study on transformer vs rnn in speech applications," in 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), December 2019, pp. 449-456.
[18] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," arXiv preprint arXiv:1704.02971, 2017.
[19] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," Physica D: Nonlinear Phenomena, vol. 404, pp. 132306, 2020.
[20] H. Ould Slimane, S. Benouadah, K. Al Shamaileh, V. Devabhaktuni, and N. Kaabouch, "ADS-B Message Injection Attack on UAVs: Assessment of SVM-based Detection Techniques," in 2022 IEEE International Conference on Electro Information Technology (eIT), May 2022, pp. 405-410.
[21] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper rnn," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 5457-5466.
[22] T. T. Khoei , H. Ould Slimane, and N. Kaabouch, “ Deep learning: Systematic review, models, challenges, and research directions. Neural Computing and Applications, “35(31), pp.23103-23124, 2023.
[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735-1780, 1997.
[24] K. Greff, R.K. Srivastava, J. Koutník, B.R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," IEEE transactions on neural networks and learning systems, vol. 28, no. 10, pp. 2222-2232, 2016.
[25] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in 2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), November 2016, pp. 324-328.
[26] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," Neural Computation, vol. 12, no. 10, pp. 2451-2471, Oct. 2000.
[27] O. Yildirim, “A novel wavelet sequence based on deep bidirectional LSTM network model for ECG signal classification,” Computers in biology and medicine, vol. 96, pp.189-202, 2018.
[28] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities,” Proceedings of the national academy of sciences, vol. 79, no. 8, pp.2554-2558, 1982.
[29] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," Neural Computation, vol. 31, no. 7, pp. 1235-1270, Jul. 2019.
[30] H. M. Lynn, S. B. Pan and P. Kim, "A Deep Bidirectional GRU Network Model for Biometric Electrocardiogram Classification Based on Recurrent Neural Networks," in IEEE Access, vol. 7, pp. 145395-145405, 2019, doi: 10.1109/ACCESS.2019.2939947.
[31] J. X. Chen, D. M. Jiang and Y. N. Zhang, "A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification," in IEEE Access, vol. 7, pp. 118530-118540, 2019, doi: 10.1109/ACCESS.2019.2936817.
[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” The journal of machine learning research, vol. 15, no. 1, pp.1929-1958, 2014.
[33] T. T. Khoei, G. Aissou, K. Al Shamaileh, V. K. Devabhaktuni and N. Kaabouch, "Supervised Deep Learning Models for Detecting GPS Spoofing Attacks on Unmanned Aerial Vehicles," 2023 IEEE International Conference on Electro Information Technology (eIT), Romeoville, IL, USA, 2023, pp. 340-346, doi: 10.1109/eIT57321.2023.10187274.
[34] T. T. Khoei, S. Ismail, K. Al Shamaileh, V. K. Devabhaktuni and N. Kaabouch, “ Impact of Dataset

and Model Parameters on Machine Learning Performance for the Detection of GPS Spoofing Attacks on Unmanned Aerial Vehicles, “ Applied Sciences, 13(1), p.383, 2022.