


## Article

# A Deep-Learning-Based GPS Signal Spoofing Detection Method for Small UAVs

Yichen Sun <sup>1,2,3</sup> , Mingxin Yu <sup>2,\*</sup>, Luyang Wang <sup>2</sup>, Tianfang Li <sup>2</sup> and Mingli Dong <sup>2,3,\*</sup><sup>1</sup> Faculty of Materials and Manufacturing, Beijing University of Technology, Beijing 100124, China<sup>2</sup> Key Laboratory of the Ministry of Education for Optoelectronic Measurement Technology and Instrument, Beijing Information Science and Technology University, Beijing 100016, China<sup>3</sup> Guangzhou Nansha Intelligent Photonic Sensing Research Institute, Guangzhou 511462, China

\* Correspondence: yumingxin@bistu.edu.cn (M.Y.); dongml@bistu.edu.cn (M.D.)

**Abstract:** The navigation of small unmanned aerial vehicles (UAVs) mainly depends on global positioning systems (GPSs). However, GPSs are vulnerable to attack by spoofing, which causes the UAVs to lose their positioning ability. To address this issue, we propose a deep learning method to detect the spoofing of GPS signals received by small UAVs. Firstly, we describe the GPS signal dataset acquisition and preprocessing methods; these include the hardware system of the UAV and the jammer used in the experiment, the time and weather conditions of the data collection, the use of Spearman correlation coefficients for preprocessing, and the use of SVM-SMOTE to solve the spoofing data imbalance. Next, we introduce a PCA-CNN-LSTM model. We used principal component analysis (PCA) of the model to extract feature information related to spoofing from the GPS signal dataset. The convolutional neural network (CNN) in the model was used to extract local features in the GPS signal dataset, and long short-term memory (LSTM) was used as a posterior module of the CNN for further processing and modeling. To minimize randomness and chance in the simulation experiments, we used the 10-fold cross-validation method to train and evaluate the computational performance of our spoofing machine learning model. We conducted a series of experiments in a numerical simulation environment and evaluated the proposed model against the most advanced traditional machine learning and deep learning models. The results and analysis show that the PCA-CNN-LSTM neural network model achieved the highest accuracy (0.9949). This paper provides a theoretical basis and technical support for spoofing detection for small-UAV GPS signals.



**Citation:** Sun, Y.; Yu, M.; Wang, L.; Li, T.; Dong, M. A Deep-Learning-Based GPS Signal Spoofing Detection Method for Small UAVs. *Drones* **2023**, *7*, 370. <https://doi.org/10.3390/drones7060370>

Academic Editor: Anastasios Dimou

Received: 24 March 2023

Revised: 20 May 2023

Accepted: 31 May 2023

Published: 2 June 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** global positioning system (GPS); spoofing; convolutional neural network (CNN); long short-term memory (LSTM); support vector machines-synthetic minority oversampling technique (SVM-SMOTE); principal component analysis (PCA)

## 1. Introduction

Small UAVs often carry multiple types of sensors, with GPS receivers being critical for small UAVs. The GPS receiver determines the location of the small UAV, as well as the spatial location and altitude at which it is located, by receiving signals from satellites. The GPS signal enables the small UAV to fly with greater precision, thus improving its mission execution capabilities. In conclusion, GPS is one of the key technologies for small UAVs. However, GPS signals are susceptible to spoofing, which can cause serious harm to GPS navigation and time synchronization systems, especially in the areas of military, aviation, navigation, and public safety uses. Countries such as Iran and India have reported security issues related to GPS signal spoofing [1–8].

To address the problem that the GPS signals of small UAVs are easily spoofed, many scholars have used multipath detection [9], energy detection [10], receiver fingerprinting [11], and machine-learning-based methods to detect the GPS signals of small UAVs. However, these approaches have certain key challenges. Multipath detection fails when

GPS signal spoofing is not obvious, and this method requires high-precision clock synchronization and additional hardware support. Energy detection loses its effectiveness with weak and noisy signals, and the method is prone to misclassifying normal signals as spoofed signals. Receiver fingerprinting requires a large amount of data for training and modeling, and has high hardware requirements for the receiver. Receiver fingerprinting is also vulnerable to environmental changes. The methods based on traditional machine learning have high requirements for data quality and feature selection. Compared with traditional machine learning methods, deep learning methods can better capture the features in the signal and extract useful information from it, and the methods can process and analyze a large amount of GPS signal data in a relatively short period of time, thus detecting GPS signal spoofing quickly.

Therefore, we propose a method for GPS signal spoofing detection based on the PCA-CNN-LSTM model. In our experimental system, the vehicle is a composite-wing UAV and the guidance system is a spoofing jammer. This method enables performance of the task of detecting whether a small UAV has been spoofed. The main contributions of this paper are as follows:

- (1) We propose a GPS signal spoofing detection model based on PCA-CNN-LSTM, which mainly consists of three parts: principal component analysis (PCA), a convolutional neural network (CNN), and a long short-term memory (LSTM) network. We chose the PCA method to downgrade the input GPS signal data, the CNN was used to extract the features of the GPS signal data, and the role of the LSTM network was to model the output of the CNN as a sequence.
- (2) To address the problem of the small amount of real GPS signal data, we propose an approach based on the SVM-SMOTE model to expand the GPS signal data.
- (3) In the simulation experimental environment, we verified that the PCA-CNN-LSTM model obtained an accuracy of 0.9943, precision of 0.9798, recall of 0.965, and an F1\_Score of 0.9722 on the GPS signal dataset.

The rest of this paper is organized as follows: In Section 2, we review related work. In Section 3, we introduce the hardware system, data acquisition, data preprocessing, and data augmentation. In Section 4, we explain the theory of PCA, the principles of the CNN and LSTM, and model training methods. In Section 5, we introduce the confusion matrix, which can be used to calculate various evaluation metrics for the model. We chose the 10-fold cross-validation method for evaluating the model performance. In addition, we give the model evaluation results. Furthermore, we compare the classification performance of this model with 15 other machine learning models. In Section 6, we summarize this study and look forward to future work.

## 2. Related Works

### 2.1. Traditional Machine-Learning-Based Methods

Meng et al. proposed a linear regression (LR) model to predict and simulate the optimal routes of UAVs. The simulation experiments showed that the LR antispoofing method can effectively enhance resistance to GPS spoofing [12]. Although the LR algorithm is simple, fast, interpretable, easy to implement and tune, and suitable for online learning, it is susceptible to outliers and vulnerable to multicollinearity, and as a linear classifier, LR has difficulty handling nonlinear relationships.

In an exception to the method based on the LR model to detect GPS signal spoofing, Shafique et al. used a machine learning (ML) algorithm to detect spoofed GPS signals using multiple GPS signal features, and the researchers generated ML models using K-fold analysis. The experimental results showed that the proposed model had an advantage in evaluating the metrics, where the accuracy was 99.00% [13]. Talaei et al. proposed two dynamic selection techniques, the metric optimized dynamic selector and the weighted metric optimized dynamic selector, to detect cyber-attacks on UAVs. The proposed methods used ten machine learning models. In the simulation state, the highest accuracy of their proposed model was 99.6% [14]. Although the use of simulation datasets can effectively

reduce the cost of data acquisition and processing, simulation datasets often cannot fully simulate the various complex situations in real scenarios, and there are various unknown situations and changes in the real world. Therefore, the models trained with simulation datasets cannot be adapted well to real scenarios, which results in performance degradation.

In addition to the flight data generated by simulation, related scholars conducted real experiments to collect data. Wei et al. proposed a detection approach called ConstDET for GPS spoofing attacks on UAVs using machine learning algorithms. The researchers conducted various real experiments to collect flight data. They selected specified types of flight data as features based on control semantics, including altitude and horizontal position control processes. The simulation experiments showed that the proposed model had a detection rate of 97.7% [15]. Nayfeh et al. proposed a machine learning model for detecting and classifying GPS spoofing in UAVs. The researchers collected and investigated static and dynamic attacks as real data and spoofing data, and performed feature analysis and data dimensionality reduction. The performance evaluation showed that the detection rate of the optimal classifier was better than 92% [16].

In addition, some researchers have attempted to use GPS and inertial measurement unit (IMU) fusion methods to detect spoofing attacks. Liang et al. proposed a lightweight active GPS spoofing detection method for UAV systems. The proposed scheme employed information fusion based on the GPS receiver and an IMU. The experiment showed that the detection rate of the method was 98.6% [8]. Feng et al. proposed an IMU, genetic algorithm (GA), and two-step XGBoost method for accurate pedestrian dead reckoning. The approach selected relevant features using GA and classified them with XGBoost. In real experiments, the model's detection rate was 96.3% (100%) correct in the hijacking (non-hijacking) cases [17]. The XGBoost algorithm has the advantages of high accuracy, scalability, robustness, and interpretability on large datasets. However, since the XGBoost algorithm needs to train multiple decision trees, it requires a large amount of data for training, and the training time of XGBoost is longer compared with that of other machine learning models.

## 2.2. Deep-Learning-Based Methods

Wang et al. proposed a new method for detecting GPS spoofing on UAVs using LSTM. In addition to GPS, this method only required existing sensor equipment. The experimental results showed that the method had a 78% detection rate [18]. Although LSTM has the ability to process sequential data, the LSTM model requires more computational resources and hands-on training, and is prone to problems such as gradient disappearance and gradient explosion.

There are also some methods to detect GPS signal spoofing through MLP methods. Shafiee et al. proposed a new approach for GPS spoofing detection using multilayer neural network (NN) algorithms. The method extracted early-late phase, delta, and signal level data to detect whether the signal was spoofed from the correlation output of the tracking loop. From the experiment, a detection rate of 99.3247% was obtained based on the NN simulation [19]. Olivia et al. proposed a software-based framework for UAV detection of GPS spoofing attacks. The researchers compared two deep learning models and found that MLP outperformed LSTM. Their approach detected GPS spoofing attacks with 83.23% (TEXBAT dataset) and 99.93% (MAVLINK dataset) accuracies [20]. Dang et al. explored the performance of a base-station (BS)-based statistical-MLP algorithm for detecting spoofing attacks on cellular-connected UAVs, which integrated spoofing attacks from six different ML models. Notably, their proposed method achieved over 97% accuracy with two BSs [21]. The MLP model is simple in structure and can achieve good performance on some simple datasets. However, MLP does not perform robustly enough for complex spoofing datasets, and thus requires a large amount of training spoofing data to avoid overfitting.

Besides the deep learning methods mentioned above, CNN-based spoofing attack detection methods have also been applied. Sung et al. proposed a deep-learning-based antispoofing method using 1D CNN as a lightweight model. The ResNet architecture was

adopted in the proposed method, which enabled it to detect most spoofed signals with better performance than support vector machines (SVMs). The algorithm's effectiveness was evaluated through flight tests [22]. Wu et al. proposed a framework for detecting cyber-attacks on UAVs using real-time sensor data. The proposed approach used a CNN-BiLSTM-Attention model. The researchers demonstrated in a simulation environment that the model was effective in identifying denial-of-service (DoS) and GPS spoofing attacks. The results showed that the model obtained 99.1% accuracy in GPS signal spoofing detection [23]. CNNs can automatically extract features and also have the advantages of parameter sharing, local connectivity, and multilayer structure, but CNNs are less suitable for sequences of GPS spoofing data and have a limited ability to capture global spoofing information.

Unlike previous work, we explain small-UAV GPS signal spoofing attack detection using the PCA-CNN-LSTM method for the first time and propose a framework that can detect small-UAV GPS signal spoofing attacks. In addition, a method of GPS signal data expansion based on the SVM-SMOTE algorithm is proposed to solve the problem of degraded computational performance of the spoofing detection model caused by the small amount of real GPS signal data. The model obtains spoofing attack detection performance with a high generalization capability and stronger robustness.

### 3. Data Acquisition and Preprocessing

In this section, we first introduce the hardware system used in this study. Next, we describe the GPS signal dataset acquisition methods, including the flight plan, flight experiment time, and weather conditions. To eliminate useless data and weakly correlated data, we used the Spearman's rank correlation coefficient to preprocess the GPS signal dataset. Finally, to improve the training and generalization of the deep learning model, we took advantage of the SVM-SMOTE method for data augmentation.

#### 3.1. Hardware System

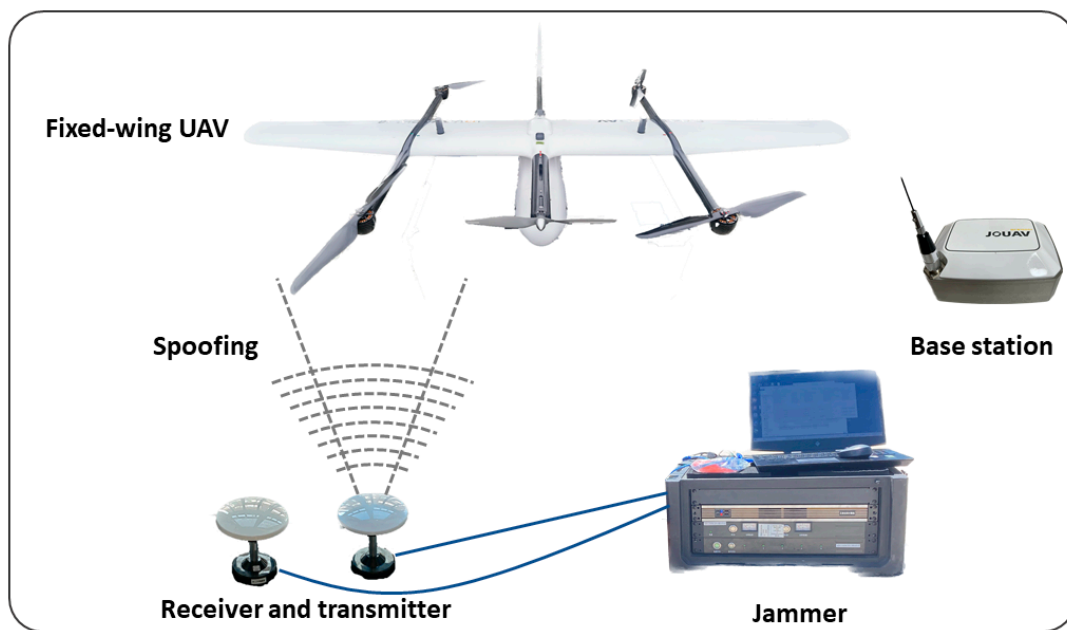
Figure 1 represents the hardware system for GPS signal spoofing detection in UAVs. The vehicle chosen for our experiments was a composite-wing UAV with a rotor and fixed wings; we chose a deceptive jammer as the instrument to spoof the GPS signal. The base station can transmit the GPS signal dataset collected by the UAV to the ground through wireless signals. The transmitter transmits the spoof signal from the jammer. The receiver receives the GPS signals in the UAV to determine the location, speed, and other information of the UAV. In the following, we elaborate on the models and specifications of the composite-wing UAV and the spoofing jammer.

This study used the CW-10 II composite-wing UAV produced by China Chengdu Zongheng Dapeng UAV Technology Co., Ltd. as the experimental aircraft. The UAV adopts a composite-wing layout consisting of a fixed wing combined with a quadcopter. This design offers the advantages of vertical takeoff and landing, which are characteristic of rotary-wing UAVs, while also considering the features of long flight time, high speed, and long distance typical of fixed-wing UAVs. Table 1 presents the technical specifications of the CW-10 II composite-wing UAV system.

**Table 1.** Technical specifications of CW-10 II composite-wing UAV system.

Projects	Main Technical Indicators
Maximum takeoff weight	12 kg
Optimal cruising airspeed	20 m/s
Maximum airspeed	30 m/s
Battery life	90 min
Wind resistance	Level 5
Takeoff and landing approach	Vertical takeoff and landing





**Figure 1.** The hardware system of the UAV GPS signal spoofing detection method.

We used a particular type of jammer, produced by China Hunan Matrix Electronic Technology Co., Ltd., as the UAV navigation guidance equipment. This equipment is based on navigation signal simulation, enhanced signal simulation, spoofing signal simulation, interference signal simulation, and real-time dynamic positioning verification. It can establish a complicated electromagnetic test environment for ground experiments. The equipment is employed to test and verify the positioning and navigation performance of GPS signals. The machine's antijamming performance includes various antijamming strategy modes, such as an antenna array, a noise ratio on the load, and signal power enhancement. Table 2 displays the primary technical indicators of a spoofing jammer.

**Table 2.** Main technical indicators of spoofing jammer.

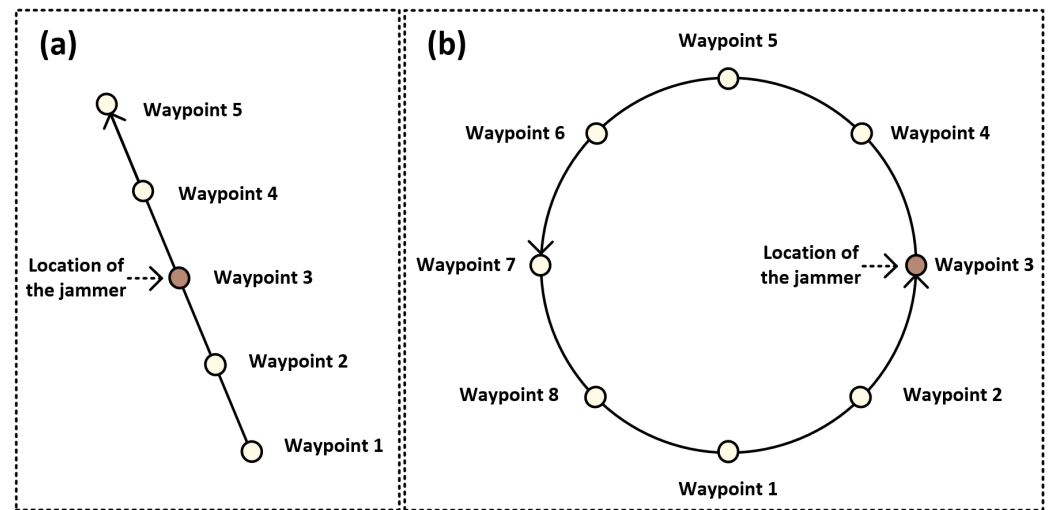
Projects	Main Technical Indicators
Spoofing signal output frequency	GPS: L1
Spoofing signal modulation mode	BPSK
Spoofing signal size	GPS: L1 16 channels
Spoofing signal power	$\leq 10$ dBm
Range	500 m
Machine power consumption	$\leq 80$ W

### 3.2. Data Acquisition

In this study, we used a CW-10 II fixed-wing UAV flight to collect flight data. In the data acquisition experiment, we could easily collect normal flight data. For the spoofed scenario, we used the jammer introduced in Section 3.1 to achieve the spoofing of GPS signals in UAVs. We set the dispersion direction of the jammer deception to due east, with a dispersion speed of 3 m/s.

Figure 2 shows the flight path diagram of the CW-10 II UAV, and we completed four kinds of flight tasks: straight flight without deception, straight flight with deception, arc flight without deception, and arc flight with deception. As shown in Figure 2a, during the UAV straight-line flight task, our UAV took off at Waypoint 1 and started deception at Waypoint 3. The deception mode was directional, driving the UAV to the east and landing it at Waypoint 5. As shown in Figure 2b, during the UAV arc flight task, our UAV took off at Waypoint 1 and started deception at Waypoint 3. The deception mode was still directional to the east, and the UAV landed at Waypoint 1 after completing a circle of the flight task.

The time and weather conditions of all the UAV flight experiments are presented in Table 3, and the flight plan is displayed in Table 4.



**Figure 2.** Flight path diagram. (a) Straight-line distance is 200 m. (b) Arc radius is 100 m.

**Table 3.** Flight test time and weather conditions of UAV.

Data	Weather Types	Temperature	Direction of Wind
27 December 2021	Sunny	−7 °C to 7 °C	South wind < Force 3
28 December 2021	Sunny	−6 °C to 5 °C	Southwest wind < Force 3
29 December 2021	Sunny	−13 °C to −5 °C	Northwest wind: Force 4–5 to 3–4
30 December 2021	Sunny	−7 °C to 6 °C	Northwest wind: Force 3–4 to < 3

**Table 4.** Flight plan of composite-wing UAV.

Flight Mode	Time of Flight	With Spoofing
Rectilinear flight	10:00:00	NO
Rectilinear flight	10:30:00	YES
Rectilinear flight	11:00:00	YES
Rectilinear flight	11:30:00	YES
Rectilinear flight	13:30:00	YES
Arc flight	14:30:00	NO
Arc flight	15:30:00	YES
Arc flight	16:30:00	YES

A total of 7699 GPS signal data (in seconds) were collected for this experiment, each with 64 features. First, we manually labeled the 7699 GPS signal data with spoofing features based on the switching times of the composite-wing UAVs and jammers in the spoofing experiments. The number of samples without spoofing (class  $S_0$ ) was 6914, which was the majority class. The number of samples with spoofing (class  $S_1$ ) was 785, which was the minority class.

### 3.3. Data Preprocessing

Once the acquisition of the GPS signal dataset was completed, we next performed the feature selection process, which included manual rejection of useless features and Spearman's rank correlation analysis.

For the features that capture redundant values in the GPS signal dataset, we performed manual rejection. We found from the features in the GPS signal dataset that since the UAV power voltage records the power level value of this UAV at a certain time, it is not targeted by spoofing attacks and is not affected by spoofing attacks. Similar features also include the

GPS recording time, recording data line number, and throttle command. There are some features that have a value of 0, such as the photo number, event, capacity, etc., regardless of whether they are subject to spoofing attacks. Therefore, all of the above features needed to be manually removed.

To analyze the relationships between the variables in the GPS signal data and make predictions based on these relationships, we used the Spearman's rank correlation coefficient to reduce the number of features. The Spearman's rank correlation coefficient is applicable to nonlinear data or in the presence of outliers, and the result falls within the interval  $[-1,1]$ . This approach can help identify potential predictors or outliers in a spoofing dataset. The formula for the Spearman's rank correlation coefficient is as follows: for the original GPS signal dataset samples, the original data  $X_i, Y_i$  are converted into rank data  $R(X), R(Y)$ , and the correlation coefficient  $\gamma_s$  is

$$\gamma_s = \rho_{R(X), R(Y)} = \frac{\text{cov}(R(X), R(Y))}{\sigma_{R(X)} \sigma_{R(Y)}} \quad (1)$$

where  $\rho$  is the Pearson product-moment correlation coefficient,  $\text{cov}(R(X), R(Y))$  is the covariance of the rank variables, and  $\sigma_{R(X)}$  and  $\sigma_{R(Y)}$  are the standard deviations of the rank variables, respectively.

Figure 3 shows the Spearman's rank correlation coefficients between features and spoofing. As can be seen from Figure 3, the Spearman's rank correlation coefficients of the feature parameters such as GPSStatus (GPS status), cmd (command), throtCmd (throttle command), latCmd (rotor lateral control command), pedCmd (rotor yaw control command), aileron (aileron command), rudder, Action (control command), and failure were less than 0.2, as they have very weak or no correlation with spoofing. As a result, we eliminated the aforementioned feature quantities that exhibit very weak or no correlation with spoofing.

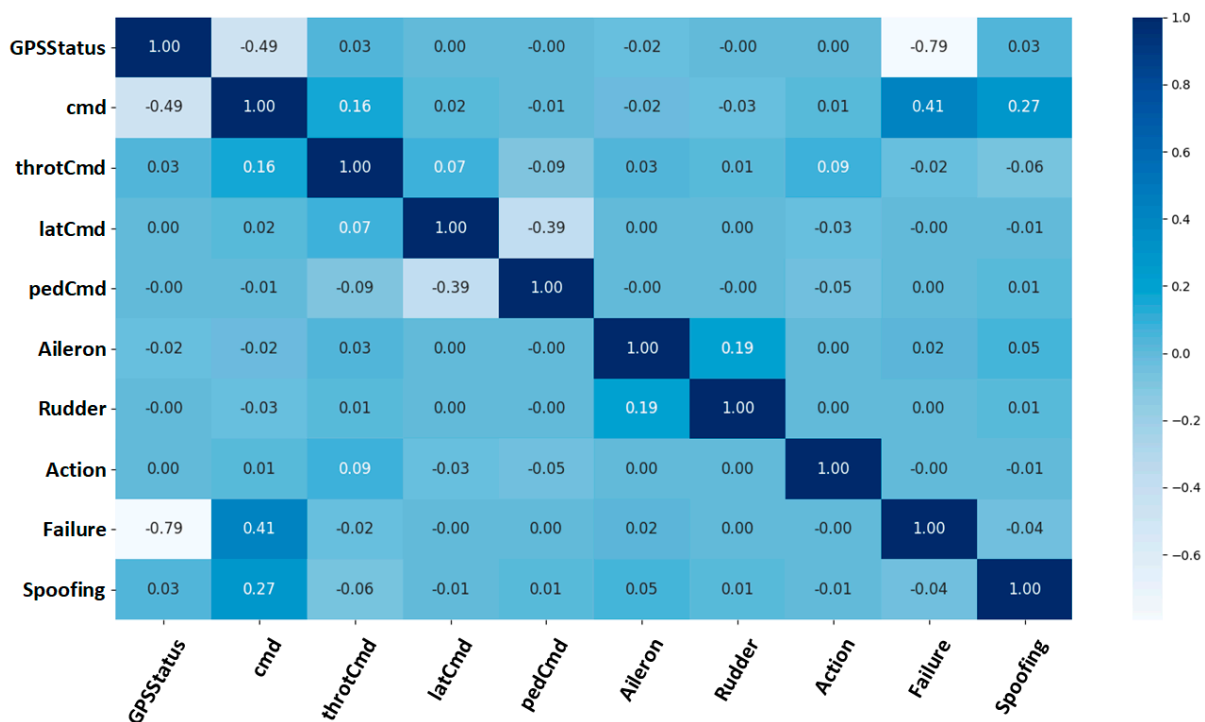


Figure 3. Spearman's rank correlation coefficient visualization of features and spoofing.

Following the Spearman's rank correlation coefficient analysis, we selected 36 features from the GPS signal dataset as input data for the model. Table 5 lists the names of the 36 features that represent the GPS signal dataset.

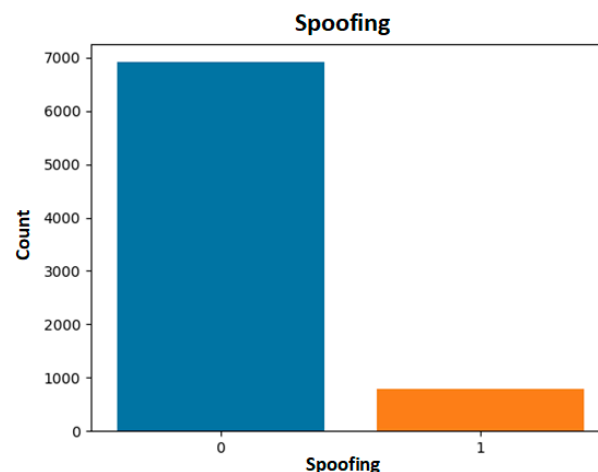
**Table 5.** Names of 36 features in GPS signal dataset.

No.	Name	Unit	No.	Name	Unit
1	GPS position accuracy	/	19	Northward ground velocity	m/s
2	GPS satellite count	/	20	Eastward ground velocity	m/s
3	GPS walk second	/	21	Downward ground velocity	m/s
4	GPS navigation solution status	/	22	Dynamic pressure	Pa
5	Avionics temperature	/	23	X magnetic field strength	milligauss
6	Latitude	deg	24	Y magnetic field strength	milligauss
7	Longitude	deg	25	Z magnetic field strength	milligauss
8	Height	meter	26	Compass	/
9	Airspeed command	/	27	Port aileron command	/
10	Euler angle of roll	deg	28	Elevator command	/
11	Euler angle of pitch	deg	29	Rudder command	/
12	Euler angle of yaw	deg	30	Longitudinal control command	/
13	X-axis acceleration	m/s <sup>2</sup>	31	X-axis acceleration deviation	m/s <sup>2</sup>
14	Y-axis acceleration	m/s <sup>2</sup>	32	Y-axis acceleration deviation	m/s <sup>2</sup>
15	Z-axis acceleration	m/s <sup>2</sup>	33	Z-axis acceleration deviation	m/s <sup>2</sup>
16	X angular velocity	deg/s	34	Pressure–height	/
17	Y angular velocity	deg/s	35	South wind	m/s
18	Z angular velocity	deg/s	36	West wind	m/s

### 3.4. Data Augmentation

After completing the GPS signal data preprocessing, we used the SVM-SMOTE [24] method for GPS signal data augmentation.

The distribution of the deception dataset according to Section 3.3 shows that the class distribution of the data is highly unbalanced, in that  $S_0$  is much more numerous than  $S_1$ . Figure 4 represents the numbers of  $S_0$  and  $S_1$ . Due to the data imbalance in the GPS signal dataset, the network overtrained  $S_0$  and did not pay enough attention to  $S_1$ , which led to the model being unreliable. Therefore, our challenge was to address the data imbalance.

**Figure 4.** The number of samples without and with spoofing.

There are various technical methods to deal with the problem of data imbalance, such as random oversampling, random undersampling, SMOTE (synthetic minority oversampling technique) [25], Borderline-SMOTE [26], SVM-SMOTE (support vector machines-SMOTE), ADASYN (adaptive synthetic sampling) [27], etc. In this study, we utilized the SVM-SMOTE method for sampling the GPS signal dataset.

SVM-SMOTE is a combination of two machine learning techniques: SVMs and SMOTE. It is used to address the problem of imbalanced datasets in classification tasks. SVM is a commonly used classifier that separates different classes of samples by constructing a hyperplane. The advantage of SVM is that it can convert nonlinear problems into

linearly separable problems by introducing kernel functions and can handle data in high-dimensional feature spaces. The SVM-SMOTE model uses the SVM algorithm to identify misclassified examples on the decision boundary. Thus, the example of the SVM-SMOTE algorithm for more synthetic spoofed samples  $\bar{S}_1$  is synthesized in regions far from the overlap with the original spoofed samples  $S_1$ ; i.e., only the  $S_1$  samples on the boundary are used to synthesize the  $\bar{S}_1$  samples, thus improving the class distribution of the samples. The advantages of the SVM-SMOTE algorithm include the following:

- The boundary information between samples is considered, and the generated synthetic samples are closer to the real minority class samples.
- Combining the advantages of SVM classifier, it can handle high-dimensional feature space and nonlinear problems.
- By increasing the number of minority category samples, the classifier's ability to recognize minority categories is improved.
- The formula for SVM-SMOTE can be expressed as follows:

We calculate the Euclidean distance  $dist(S_0, S_1)$  between the spoofing data  $S_0$  and the data without spoofing  $S_1$ . For each minority sample,  $S_1$ , we find its  $k$  nearest neighbors,  $N(S_0)$ :

$$N(S_1) = \{S_0 | dist(S_0, S_1) \text{ is minimum in minority samples} \} \quad (2)$$

We compute the synthetic samples,  $\bar{S}_1$ , by interpolating between  $S_1$  and a randomly selected neighbor,  $S_0$ :

$$\bar{S}_1 = S_1 + rand(0, 1) \times (S_0 - S_1) \quad (3)$$

We filter the synthetic samples by removing any that fall into the majority class region. This can be achieved by checking whether the synthetic sample falls into the convex hull of the majority class samples. We train an SVM using the original minority class samples and the synthetic samples generated in Equation (3), and we use the trained SVM to classify the test data.

The most important parameters in the SVM-SMOTE resampling method used in this study are set as follows: K\_Neighbors = 5 and M\_Neighbors = 10. Using these settings, the model achieves the best results. After we use SVM-SMOTE, the number of spoofing samples in the GPS signal dataset increases to the same number as samples without spoofing. Thus, the total number of GPS signal datasets  $\bar{S}$  after resampling is 13,828.

To make different features comparable in  $\bar{S}$  after data enhancement and to improve the convergence speed and accuracy of the algorithm, we normalize in  $\bar{S}$ . Normalization is a data preprocessing technique that scales the range of values of the input data to [0,1]. The equation of normalization is as follows:

$$\bar{S}^* = \frac{\bar{S} - \mu}{\sigma} \quad (4)$$

where  $\mu$  is the mean of all  $\bar{S}$  and  $\sigma$  is the standard deviation of all  $\bar{S}$ .

#### 4. Methods

This section provides an explanation of the PCA-CNN-LSTM model and its working mechanism, as well as an introduction to its training method. Figure 5 illustrates the structure of the proposed PCA-CNN-LSTM network. In the model, PCA is used as a preprocessing step to reduce the dimensionality of the GPS signal dataset and identify the most important features. The dimensions of the GPS signal dataset are further reduced from 36 to 15 dimensions after processing. The CNN is then used to extract relevant features from the reduced GPS signal dataset, while LSTM is used to model the temporal dependencies in the data. By combining these techniques, the model can detect the output spoof value (0 or 1) and determine whether the received GPS signal is spoofed or not. In the next section, we describe in detail the roles of each module in the PCA-CNN-LSTM.



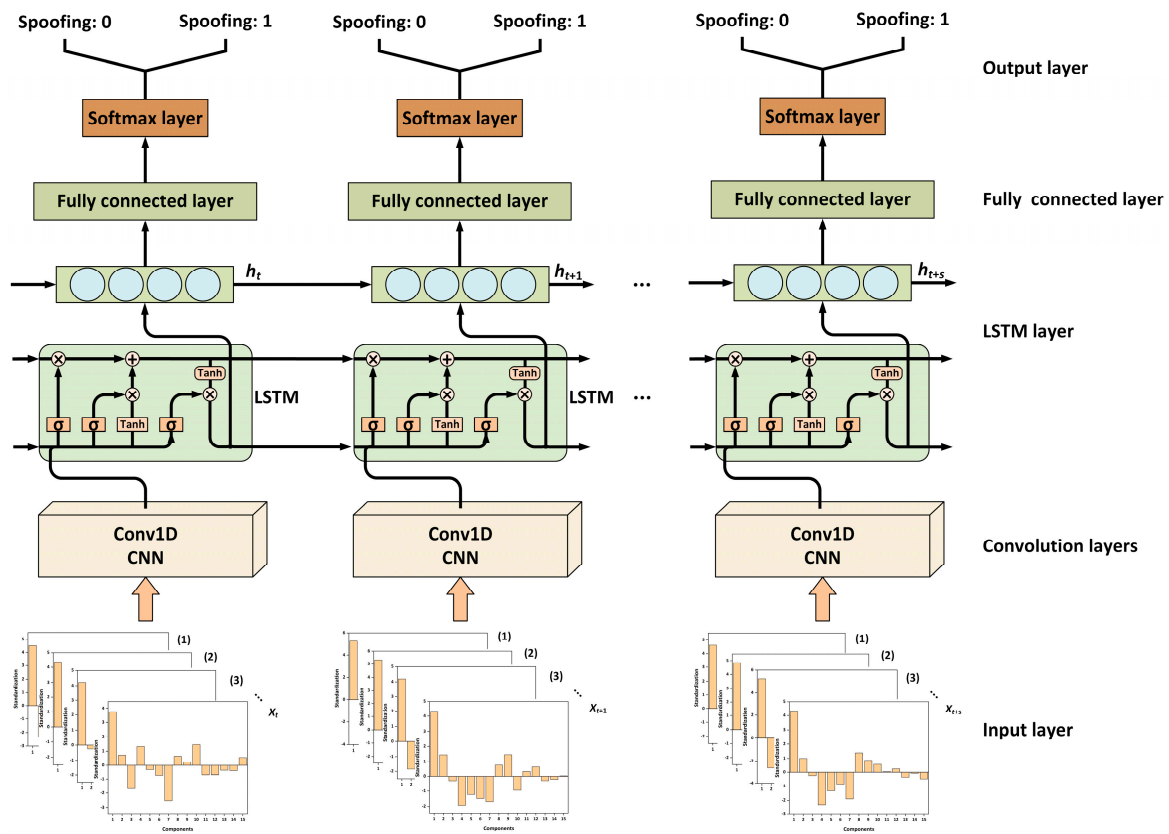


Figure 5. PCA-CNN-LSTM network structure.

#### 4.1. PCA

According to Section 3.3, it is known that 36-dimensional features still existed in the deception data. In order to extract the effective feature dimension information, we used the PCA method for data dimensionality reduction [28].

There were 13,828 samples  $\{X_1, X_2, \dots, X_{13828}\}$  in the GPS signal dataset, and each sample had 36-dimensional features  $X_i = (x_1^i, x_2^i, \dots, x_{36}^i)^T$ , each feature being  $x_j$  with its own feature value.

All features were first centralized, i.e., deaveraged, and the mean value of each feature in the dataset is shown below.

$$\bar{x}_j = \frac{1}{13828} \sum_{i=1}^{13828} x_j^i \quad (5)$$

where  $j$  denotes the  $j$ -th feature. After the deaveraging process, the values of the original GPS signal dataset features became the new feature values. Next we found the covariance matrix  $C$ , which is the covariance matrix of these 13,828 samples under these 36-dimensional features:

$$C = \begin{bmatrix} \text{cov}(x_1, x_1) & \text{cov}(x_1, x_2) & \dots & \text{cov}(x_1, x_{36}) \\ \text{cov}(x_2, x_1) & \text{cov}(x_2, x_2) & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \text{cov}(x_{36}, x_1) & \dots & \dots & \text{cov}(x_{36}, x_{36}) \end{bmatrix} \quad (6)$$

where the diagonal in  $C$  is the variance from  $x_1$  to  $x_{36}$ , and the off-diagonal is the covariance. The larger the absolute value of the covariance, the greater the influence of the two on each other. The covariance equation  $\text{cov}(x_m, x_m)$  is as follows:

$$\text{cov}(x_m, x_m) = \frac{\sum_{i=1}^{13828} (x_m^i - \bar{x}_m)(x_m^i - \bar{x}_m)}{13828 - 1} \quad (7)$$

where  $m$  is the variance value of the  $m$ -th feature. Then we found the eigenvalues of  $\mathbf{C}$  and the corresponding eigenvectors  $\mu$ , since each eigenvalue corresponds to an eigenvector:

$$\mathbf{C}\mu = \lambda\mu \quad (8)$$

where  $\lambda$  is the eigenvalue, there are 36 eigenvalues  $\lambda$ , and each  $\lambda_m$  corresponds to an eigenvector  $\mu_m$ . The eigenvalues  $\lambda$  are sorted in descending order, and the top  $k$  largest ones are selected, and their corresponding  $k$  eigenvectors are listed. We obtain a set:  $\{(\lambda_1, \mu_1), (\lambda_2, \mu_2), \dots, (\lambda_k, \mu_k)\}$ . The process of selecting the top  $k$  largest eigenvalues and the corresponding eigenvectors and projecting them is the process of dimensionality reduction. The new features are calculated as follows:

$$\begin{bmatrix} y_1^i \\ y_2^i \\ \vdots \\ y_k^i \end{bmatrix} = \begin{bmatrix} \mu_1^T \cdot (x_1^i, x_2^i, \dots, x_{36}^i) \\ \mu_2^T \cdot (x_1^i, x_2^i, \dots, x_{36}^i) \\ \vdots \\ \mu_k^T \cdot (x_1^i, x_2^i, \dots, x_{36}^i) \end{bmatrix} \quad (9)$$

The variance contribution of the  $j$ -th principal component  $Y_j$  is as follows:

$$Y_k = \frac{\lambda_k}{\sum_{i=1}^{36} \lambda_i} \quad (10)$$

The contribution margin is the proportion of the variance of the principal component to the total variance of the original data, and the larger the contribution margin, the stronger the ability of the principal component to explain the total variance. Thus, the cumulative contribution rate of the first  $m$  principal components  $Y_m$  is as follows:

$$Y_m = \frac{\sum_{i=1}^m \lambda_i}{\sum_{i=1}^{36} \lambda_i} \quad (11)$$

Using Equations (5)–(11), we performed dimensionality reduction on the remaining 36 features in the GPS signal dataset using the PCA algorithm. Figure 6 displays the contribution margin of the principal components in deception spoofing across the 36 dimensions. The contribution of the first 15 components was 0.9136; thus, we selected 15 as the number of primary components in the GPS signal dataset.

#### 4.2. CNN

A CNN is a type of artificial neural network that is specialized in processing data with a lattice-like topology. In CNNs, the layers are designed to perform convolution operations on the input data, followed by pooling operations to reduce the spatial dimensions of the data. Then, the resulting data are passed through fully connected layers to produce an output. Convolution layers are responsible for learning local patterns in the input data, while the fully connected layers are responsible for combining the information from all the local patterns to make a prediction. For a GPS signal dataset that was arranged in chronological order, we used a one-dimensional (1D) convolutional structure. Figure 7a shows the schematic of the 1D convolutional operation, where the convolution kernel is 3 and the three colors (red, green, and blue) represent the three convolution kernels.

We used the PReLU activation function to process the convolution results in this study. The PReLU function is an activation function used in artificial neural networks. It is an extension of the widely used ReLU function with the addition of learnable parameters that allow the function to have a negative slope [29]. The PReLU function is defined as follows:

$$\phi_{\text{PReLU}} = \max(\alpha x, x) \quad (12)$$

where  $\alpha$  is a learnable parameter that is determined during the training process. The negative slopes in PReLU allow the activation function to have more flexibility and can help to avoid the dying ReLU problem, where neurons can become inactive during the training process. This can, in turn, improve the model's ability to fit the data and increase its overall performance.

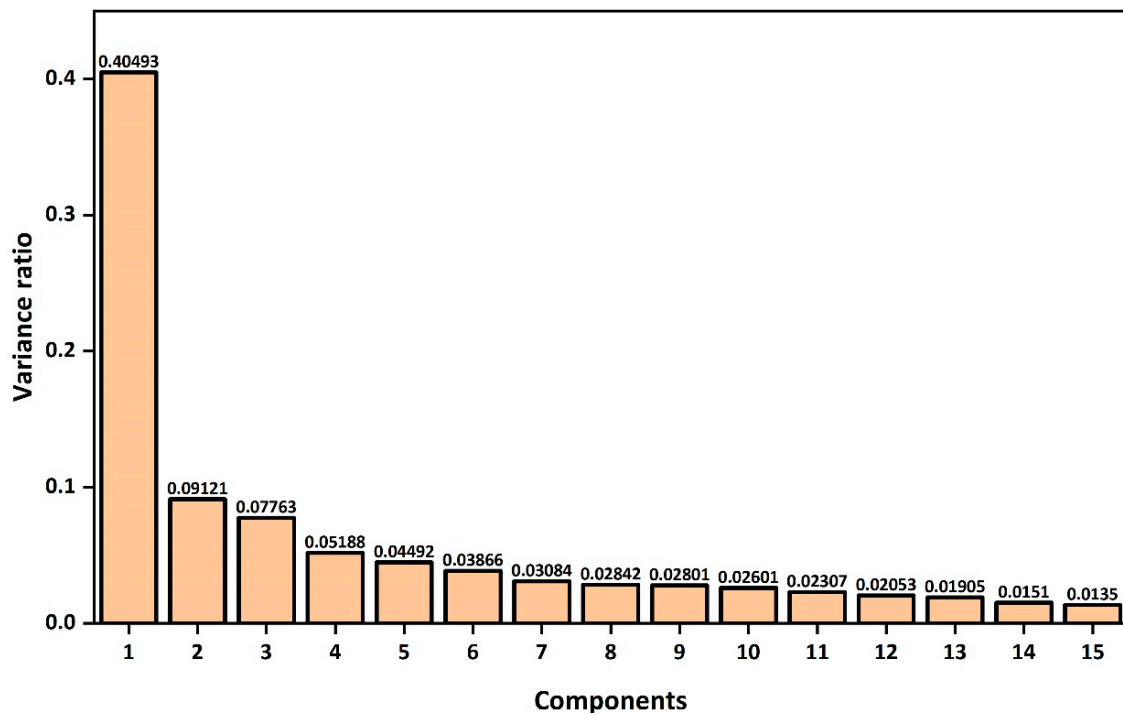


Figure 6. Principal component contribution margin of 36-dimension spoofing.

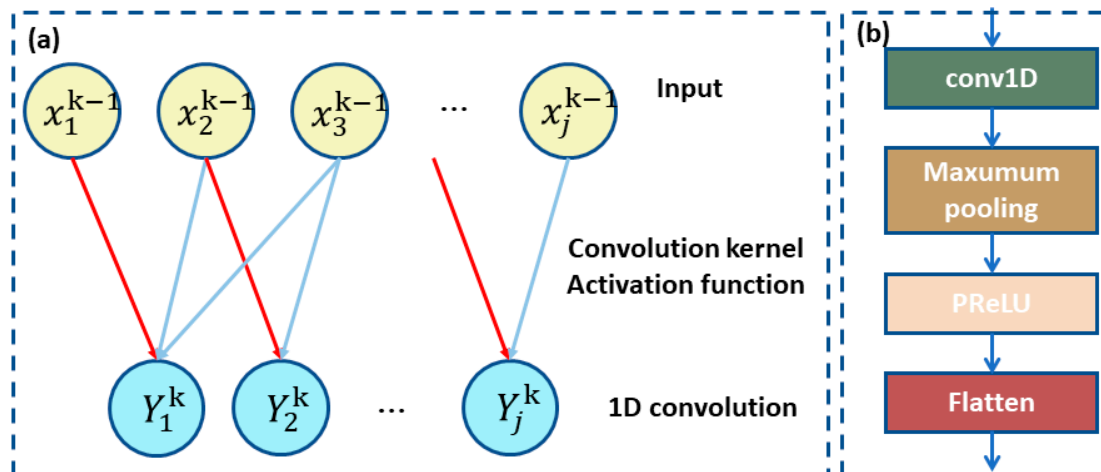


Figure 7. (a) One-dimensional convolutional operation schematic. (b) The network structure of CNN.

Figure 7b illustrates the CNN network structure. The pooling layer in CNNs performs downsampling by reducing the spatial dimensionality of the feature map. The primary goal of this layer is to make the feature map representation smaller and more manageable. This has two key advantages: First, it reduces the number of parameters and computations in the network, making it faster and more efficient. Second, it helps prevent overfitting by providing some translation invariance, which means that small changes in the input do not lead to significant changes in the output. Different types of pooling operations can be performed, including max pooling, average pooling, and sum pooling. In our work,

we used maximum pooling, which effectively reduces the dimensionality of the data and improves the robustness of the results.

The output of the maximum pooling is represented as follows:

$$X = \max(x_1, x_2, x_3, \dots, x_n) \quad (13)$$

where  $X$  denotes the output value of the largest pool and  $x_i$  denotes the value of each pool.

After multiple convolutional layers, activation function layers, and pooling layers, features are extracted from the input information and fed to the LSTM network for prediction. Therefore, the feature  $Y$  is defined as follows:

$$Y = \text{PReLU} \left( \text{PReLU} \left( \dots \text{PReLU} \left( b_j^k + \sum_{i=1}^{N_{k-1}} \text{conv1D}(x_i^{k-1}, w_{ij}^{k-1}) \right) \right) \right) \quad (14)$$

where  $x_i^{k-1}$  is the input value of layer  $k-1$ ,  $w_{ij}^{k-1}$  is the convolution kernel,  $b_j^k$  is the bias of layer  $k$ ,  $N_{k-1}$  is the set of input features, and  $\text{conv1D}$  is used for 1D convolution.

#### 4.3. LSTM

LSTM is a commonly used recurrent neural network (RNN) model for processing sequential data. It has three gates (input gate, forget gate, and output gate) and a cell state to control the flow and preservation of information.

Figure 8 represents the LSTM network structure. The forget gate of LSTM is used to control the information that needs to be forgotten in the previous moment's cell state. The inputs to the forget gate are the previous moment's memory state and the current moment's input. The input gate is used to control what needs to be added to the memory state at the current moment. The inputs to the input gate are the memory state of the previous moment and the input of the current moment. The output gate is used to control the information that needs to be output from the memory state at the current moment. Cell state is the long-term memory unit of LSTM, which is used to store the historical information of sequence data. At each moment, the update operation of forget gate, input gate, and cell state can control the information retained, added, and forgotten in the memory state at the current moment. The hidden state is the short-term memory unit of LSTM, which is used to store the current information of sequence data. At each moment, the output gate can control the information output in the hidden state.

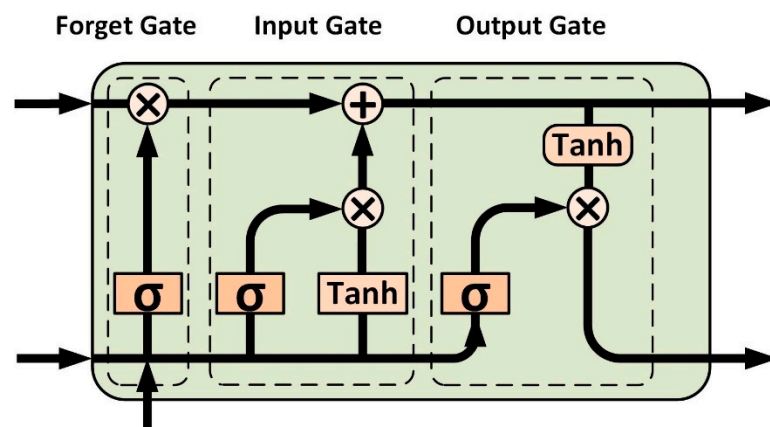


Figure 8. LSTM network structure diagram.

The calculation methods of the input gate  $i_{t+1}$ , forget gate  $f_{t+1}$ , output gate  $o_{t+1}$ , cell gate  $c_{t+1}$ , and hidden gate  $h_{t+1}$  are as follows:

$$i_{t+1} = \text{Sigmoid}(W_i[h_t, x_{t+1}] + b_i) \quad (15)$$

$$f_{t+1} = \text{Sigmoid}(W_f[h_t, x_{t+1}] + b_f) \quad (16)$$

$$o_{t+1} = \text{Sigmoid}(W_o[h_t, x_{t+1}] + b_o) \quad (17)$$

$$c_{t+1} = f_{t+1} \bullet c_t + i_{t+1} \bullet z_{t+1} \quad (18)$$

$$h_{t+1} = o_{t+1} \bullet \text{Tanh}(c_{t+1}) \quad (19)$$

where  $W_i$ ,  $W_f$ , and  $W_o$  represent the weighting matrix of input gate  $i_{t+1}$ , forget gate  $f_{t+1}$ , and output gate  $o_{t+1}$ , respectively, and  $b_i$ ,  $b_f$ , and  $b_o$  represent the learning bias in the training process.  $h_t$  represents the hidden state in the model, and  $[h_t, x_{t+1}]$  represents the longer vector in the connection of the two vectors.

#### 4.4. PCA-CNN-LSTM Model Training

We determined the optimal combination of parameters for the PCA-CNN-LSTM model using ablation experiments. The optimal combination of these combined models is shown in Table 6, which includes the number of filters, kernel size, and activation function for each convolutional layer, the pool size for the pooling layer, and the total number of parameters for all layers, including the LSTM layer. Depending on the characteristics of the learned data, adjusting these parameters affects the speed and efficiency of learning. After PCA dimensionality reduction, the input data of the network are  $\{x_1, x_2, \dots, x_{15}\}$ . First, we input the reduced-dimensional GPS signal spoofing dataset into the 1D convolutional layer, which is used to extract the spatial features of the input data. The data are then fed into the LSTM network, which serves to model the timing characteristics of the GPS signal spoofing data. With the combination of these three modules, the PCA-CNN-LSTM model is able to better process data from GPS signal spoofing interference detection and extract useful features for prediction or classification tasks.

**Table 6.** Configuration of each layer of PCA-CNN-LSTM model.

Proposed Model		Value
Convolution	Filters	8
	Kernel_size	1
	Activation function	PReLU
Maxpooling	Pool_size	1
	Activation function	PReLU
LSTM	Hidden nodes	50
	Activation function	Tanh
TimeDistributed	Dense	64
	Activation function	PReLU
Output	—	2

In the training of PCA-CNN-LSTM model, cross-entropy function was used as the loss function of the model. Cross entropy is an important concept in information theory, which is mainly used to measure the difference between two probability distributions. The formula of cross-entropy function is as follows:

$$H(p, q) = - \sum_{i=1}^n p(x_i) \log(q(x_i)) \quad (20)$$

where  $p(x_i)$  stands for the actual spoofing, and  $q(x_i)$  stands for the spoofing predicted by the model.

Next, we introduced the hyperparameter selection of PCA-CNN-LSTM model. We used 30 epochs and a batch size of 2 in the model. Early stopping was employed, and



a portion of the training set was used as the validation set. The validation set was used to monitor the convergence during training and to determine whether the model should be stopped early based on convergence changes. The Adam optimizer was used in the PCA-CNN-LSTM model with a learning rate of 0.001. We used exponential decay learning rate as a learning rate adjustment strategy in training neural networks. The loss function utilized the cross-entropy function.

The PCA-CNN-LSTM model was developed using the Python (version 3.6.13) and Keras (version 2.4.3, Google Inc.) frameworks. The model was trained on a desktop computer equipped with an NVIDIA RTX 3090 graphical processing unit (GPU), an Intel (R) Core (TM) i9-10900K CPU running at 3.70 GHz, and 64 GB of RAM. The computer was running the Windows 10 operating system (Microsoft).

## 5. Results

We first present the evaluation method of the computational model, namely the confusion matrix and its secondary and tertiary indicators. Next, we describe a sliding-window-based 10-fold cross-validation method that we used to train the model and evaluate its computational performance. We then present the evaluation results of the GPS signal spoofing detection model based on the PCA-CNN-LSTM neural network and compare them with those of other machine learning and deep learning models.

### 5.1. Evaluation Method

The confusion matrix is an index used to evaluate classification models. Through sample collection, the true positive and true negative values of the real and predicted labels can be determined. Table 7 represents the binary confusion matrix. The first-level indicators are as follows:

- True positive (TP): The number of cases in the model where the true and predicted labels are both positive.
- False negative (FN): The number of cases in the model where the true label is positive but the predicted label is negative.
- False positive (FP): The number of cases in the model where the true label is negative but the predicted label is positive.
- False negative (TN): The number of cases in the model where the true and predicted labels are both negative.

**Table 7.** Binary confusion matrix.

Confusion Matrix		Predicted Label	
		Negative	Positive
True label	Negative	TN	FN
	Positive	FP	TP

Next, we introduce the secondary indexes in the confusion matrix. Accuracy is the ratio of the number of correct predictions in a classification model to the total number of predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (21)$$

Precision refers to the percentage of all predicted labels that are positive and correct.

$$Precision = \frac{TP}{TP + FP} \quad (22)$$

Recall, also known as sensitivity, refers to the proportion of the model's predicted labels that are positive and correct to the total number of actual positive labels.

$$Recall = \frac{TP}{TP + FN} \quad (23)$$

F1\_Score is the harmonic average of precision and recall, which is a tertiary index in the confusion matrix. It takes into account both precision and recall, favoring smaller values and penalizing extreme cases where accuracy and recall differ greatly.

$$F1\_Score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (24)$$

### 5.2. Ten-Fold Cross-Verification

To minimize randomness and chance in the simulation experiments, we used the 10-fold cross-validation method to train and evaluate the computational performance of our spoofing machine learning model.

As shown in Figure 9, we first divided the GPS signal dataset, which had undergone data enhancement, into 10 mutually exclusive parts using hierarchical sampling. During each training iteration, we selected one part of the dataset as the test set and the remaining parts as the training set. We then applied the SVM-SMOTE algorithm to expand the data in the training set using both the SMOTE and SVM methods. For example, during the first training iteration, we used A1 as the test set and A2–A10 as the training set, and obtained the new test set C1 and training set B1 through data augmentation. This process was repeated for each training iteration, until the tenth iteration, where we used A10 as the test set and A1–A9 after data augmentation as the training set, and obtained the new test set C10 and training set B10 through data augmentation.

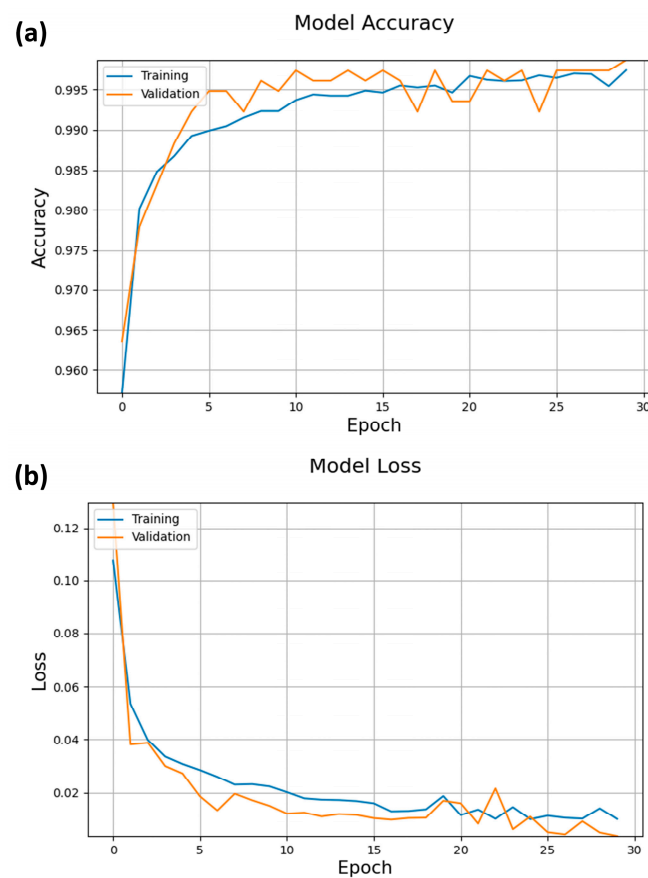


Figure 9. Ten-fold cross-validation structure.

Next, we used the training sets B1–B10 after data enhancement to train the model, and evaluated the resulting training models using the original A1–A10 datasets before data enhancement. We also used the original test set before data enhancement to ensure the authenticity of the model evaluation results.

### 5.3. Model Evaluation Results

The accuracy and loss functions of the PCA-CNN-LSTM model are shown in Figure 10. From Figure 10, we can find that because we used early stopping, the validation error of the PCA-CNN-LSTM model did not improve when the epoch was 29, and the model reached the optimal point and stopped training. Table 8 indicates the evaluation results of the PCA-CNN-LSTM model after the 10-fold cross-validation test. As shown in Table 8, the PCA-CNN-LSTM model has an average accuracy of 0.9943, an average precision of 0.9798, an average recall of 0.965, and an average F1\_Score of 0.9722. In the experiments from fold 1 to fold 10, accuracy was in the range (0.9896, 0.9987), precision was in the range (0.9306, 1), recall was in the range (0.939, 0.9889), and the F1\_Score was in the range (0.9437, 0.9944). Moreover, in the 1st, 2nd, 5th, and 10th fold experiments, the model obtained a precision of even up to 1 on the GPS signal dataset.



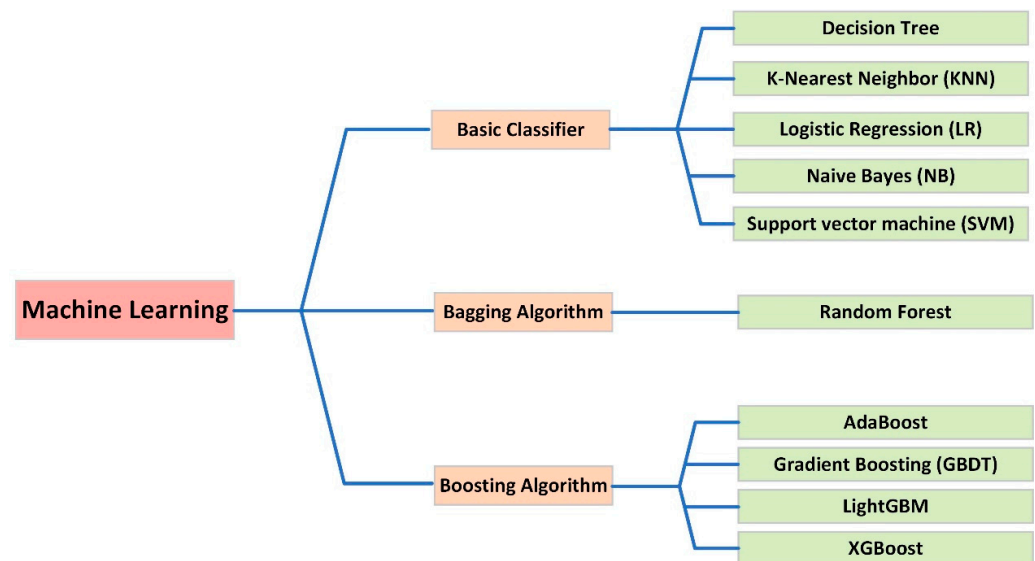
**Figure 10.** Accuracy (a) and loss (b) functions of PCA-CNN-LSTM model.

**Table 8.** Evaluation results of PCA-CNN-LSTM model after 10-fold cross-validation test.

	Accuracy	Precision	Recall	F1_Score
1	0.9987	1	0.9889	0.9944
2	0.9935	1	0.939	0.9686
3	0.9948	0.974	0.974	0.974
4	0.9961	0.9863	0.973	0.9796
5	0.9974	1	0.9661	0.9828
6	0.9922	0.9659	0.9659	0.9659
7	0.9896	0.9306	0.9571	0.9437
8	0.9896	0.9659	0.9444	0.9551
9	0.9961	0.9756	0.9877	0.9816
10	0.9948	1	0.9535	0.9762
Means	0.9943	0.9798	0.965	0.9722

#### 5.4. Comparison with Traditional Machine Learning Models

After completing data enhancement and feature selection on the GPS signal dataset, we established a GPS signal spoofing detection model based on machine learning. We selected various algorithms to establish the model, including basic classifiers, such as decision trees, the K-nearest neighbor (KNN), logistic regression (LR), naive Bayes (NB), and SVM algorithms, bagging algorithms (using the random forest (RF) integrated learning algorithm), boosting algorithms (including the AdaBoost algorithm, Gradient Boosting (GBDT) algorithm, LightGBM algorithm, and XGBoost algorithm), and neural networks. Figure 11 shows the machine learning model of GPS signal spoofing detection.



**Figure 11.** Establishment of machine learning model of GPS signal spoofing detection.

The machine learning model of GPS signal spoofing detection has many hyperparameters that need to be manually selected in advance. Improper selection of hyperparameters may lead to overfitting or underfitting of the model. Therefore, there are two ways to select hyperparameters: one is to select hyperparameters of different sizes, bring them into the model, and select the parameters with the best performance; the other is to fine-tune the model hyperparameters empirically. However, manual tuning is lengthy and time-consuming. Therefore, we adopted the grid search method to modulate the hyperparameters of various machine learning models in this study. Grid search adjusts parameters in sequence according to the set step size in the specified parameter range and uses the adjusted hyperparameters to train the machine learning model to find the hyperparameter with the highest accuracy in the verification set. Therefore, grid search is a process of training and comparison. Table 9 shows the hyperparameter types and values of the supervised machine learning model used to detect GPS signal spoofing after grid search and 10-fold cross-validation training were adopted in this study.

#### 5.5. Comparison with Other Deep Learning Models

We selected the PCA-MLP and PCA-CNN as well as the PCA-LSTM and PCA-GRU [30] neural networks as GPS signal spoofing detection models. We compared their computational performance with that of the PCA-CNN-LSTM model proposed in this paper. The hyperparameter selection of the four neural network models mentioned above was consistent with the PCA-CNN-LSTM model. Figures 12–15 show the accuracy and loss functions of the four neural network models. In particular, after using the early stop method, the epochs of PCA-LSTM, PCA-GRU, PCA-MLP, and PCA-CNN-GRU are 15, 23, 14, and 23, respectively.

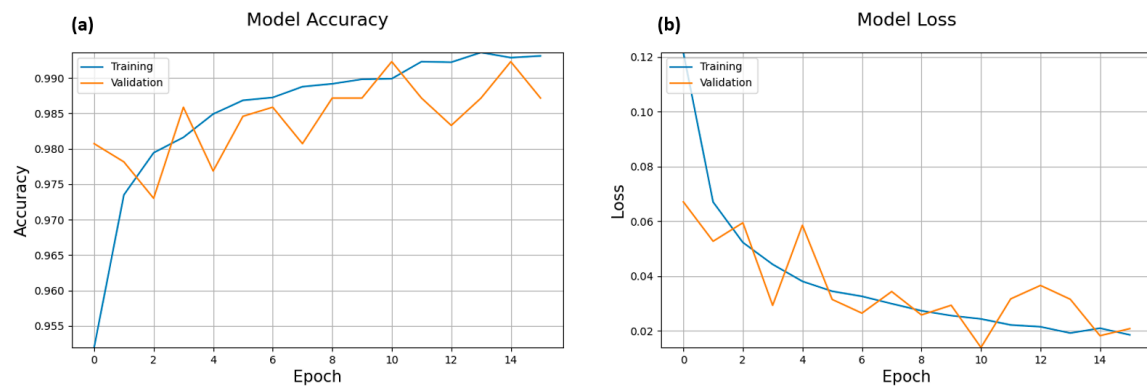


Figure 12. Accuracy (a) and loss functions (b) of the PCA-LSTM model.

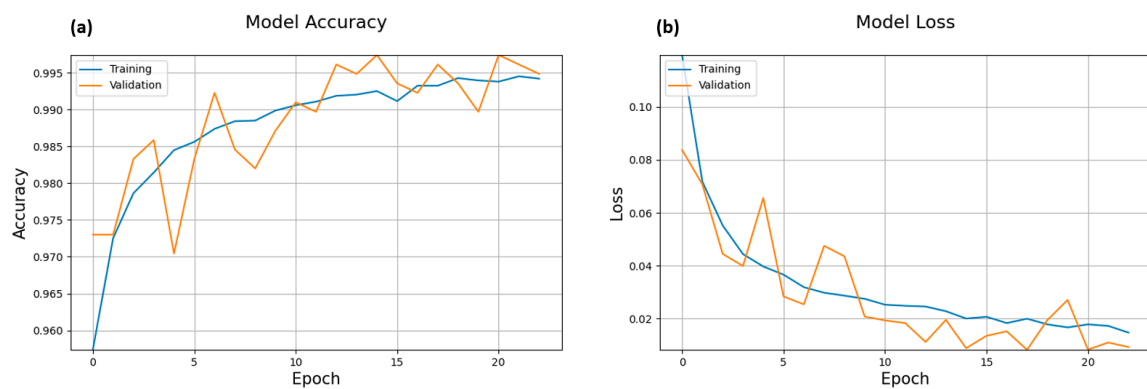


Figure 13. Accuracy (a) and loss functions (b) of the PCA-GRU model.

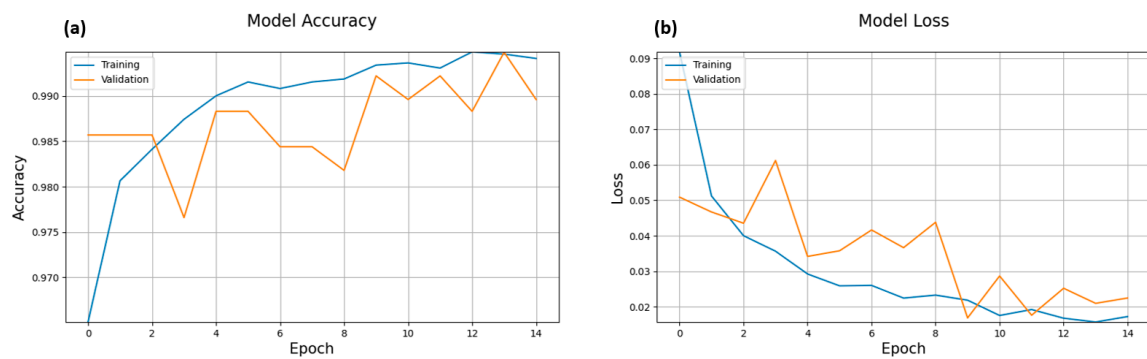


Figure 14. Accuracy (a) and loss functions (b) of the PCA-MLP model.

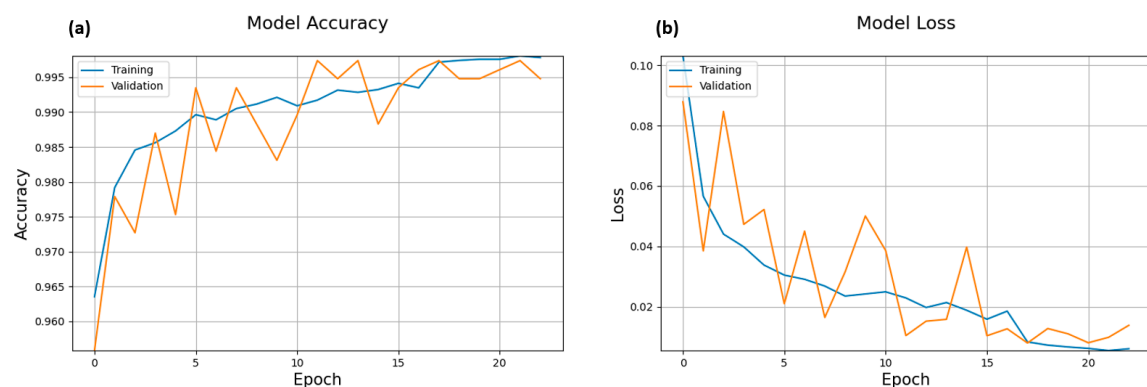


Figure 15. Accuracy (a) and loss functions (b) of the PCA-CNN-GRU model.



**Table 9.** Types and values of hyperparameters of machine learning model for GPS signal spoofing detection.

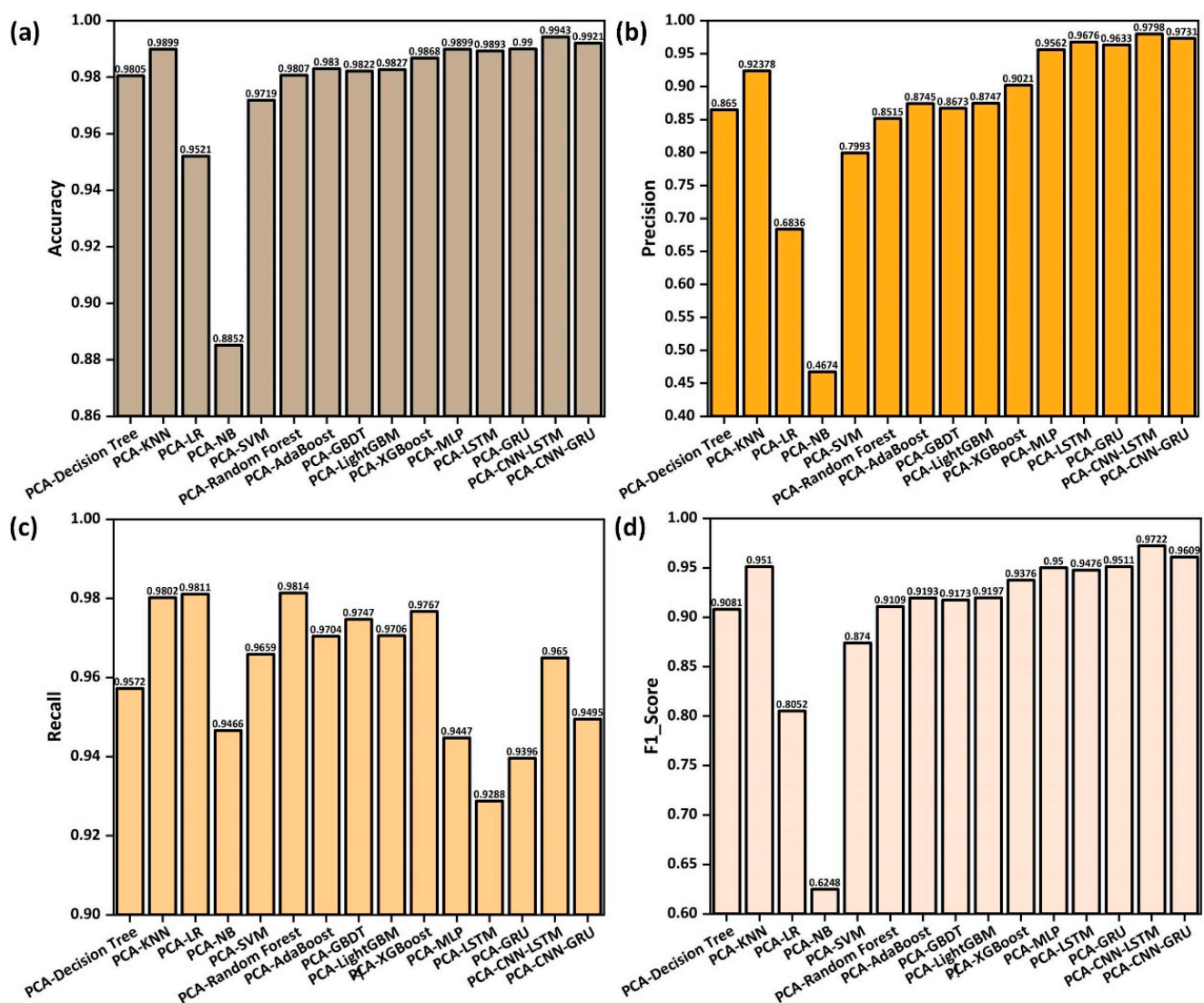
Model	Hyperparameter		
<b>PCA-Decision Tree</b>	<b>Max_depth = 5</b>		min_samples_leaf = 5
	Max_leaf_node = 30		Criterion = entropy
PCA-KNN	N-neighbor = 9	P = 5	Weights = Uniform
PCA-LR	C = 10		Penalty: L2
PCA-NB		/	
PCA-SVM	C = 10	Kernel = rbf	Gamma = 0.01
PCA-Random Forest	C = 10	Kernel = rbf	Gamma = 0.01
PCA-AdaBoost	Criterion = entropy	Learning_rate = 0.001	N_estimators = 50
	Max_leaf_nodes = 25	Max_depth = 20	
PCA-GBDT	Learning_rate = 0.05	Max_depth = 10	min_samples_leaf = 70
	Min_sample_split = 100	N_estimators = 50	Subsample = 0.75
PCA-LightGBM	Learning_rate = 0.005	Max_depth = 5	min_samples_leaf = 60
	Min_sample_split = 300	N_estimators = 800	Subsample = 0.6
	Max_depth = 5	Min_child_weight = 5	Min_sample_split = 100
PCA-XGBoost	N_estimators = 200	Subsample = 0.85	Reg_alpha = 0
		Learning_rate = 0.025	

We tested the supervised machine learning model of GPS signal spoofing on an expanded and divided GPS signal dataset using data enhancement, grid search, and 10-fold cross-validation. Figure 16 shows the accuracy, precision, recall, and F1\_Score obtained by the 15 models selected in Sections 5.3–5.5 on the GPS signal dataset. The PCA-CNN-LSTM model proposed in this paper had the highest accuracy, precision, and F1\_Score values among the 15 models for detecting GPS signal spoofing, which were 0.9943, 0.9798, and 0.9722, respectively. The recall detected by the PCA-Random Forest model was the highest at 0.9814. According to the performance index, our proposed PCA-CNN-LSTM model had better classification performance in GPS signal spoofing detection.

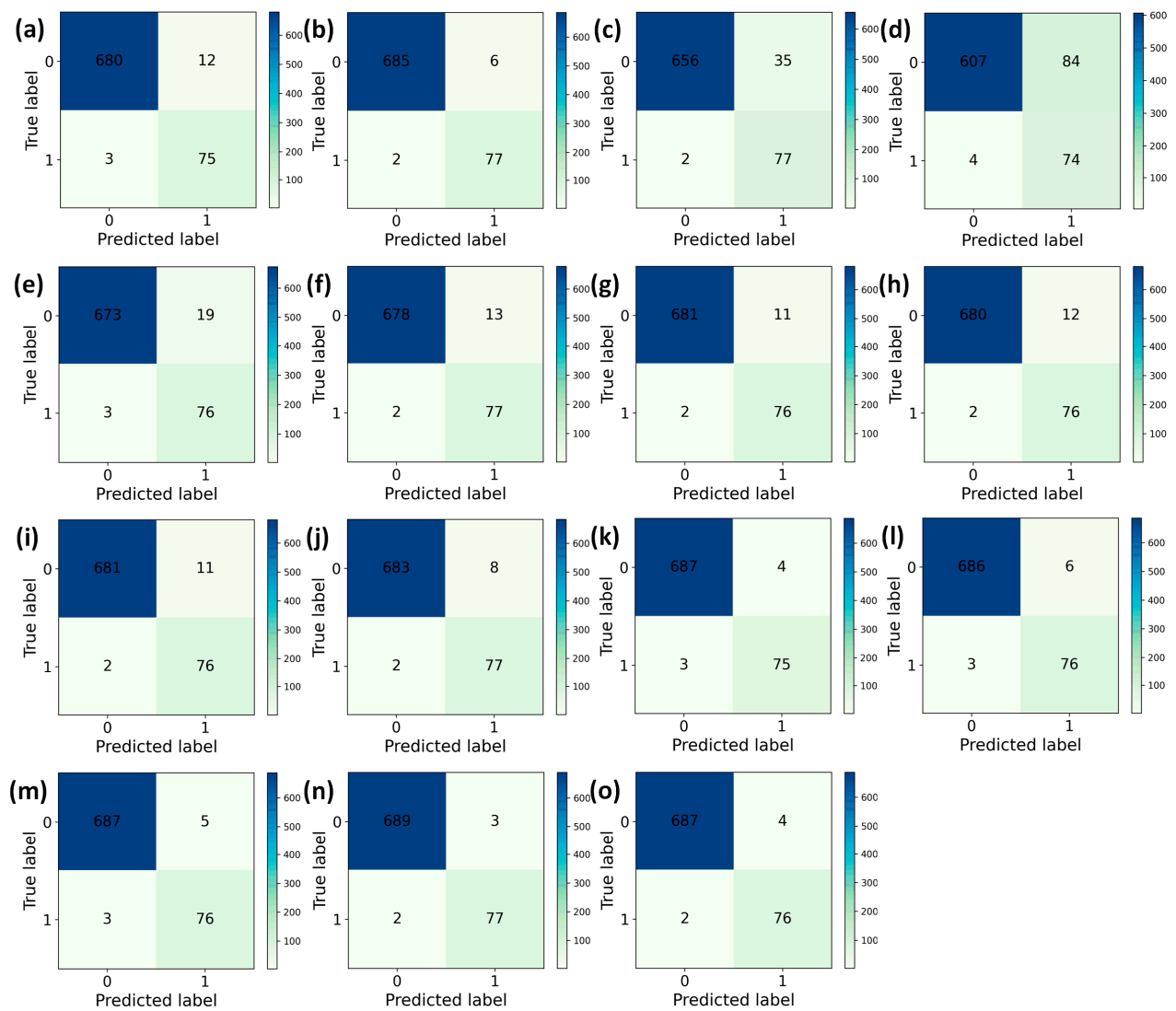
Figure 17 represents the average confusion matrix plot for various deception models on the deception dataset. Among them, Figure 17n (PCA-CNN-LSTM) has the highest number of primary diagonal elements TN and TP with 689 and 77, respectively, which indicates that the model classifies the highest number of correct samples. Moreover, the secondary diagonal elements FN and FP in Figure 17 have the smallest numbers of 3 and 2, respectively. This also verifies that our proposed PCA-CNN-LSTM model has better inference performance for GPS signal spoof detection compared with other machine learning models.

Based on the quantitative analysis of the 15 machine learning models, it is clear that the four evaluation metrics obtained by the deep learning models on the UAV GPS signal spoofing dataset are somewhat higher compared with the traditional machine learning models. This is because traditional machine learning usually requires manual selection and extraction of features, while deep learning models can automatically learn to extract features from raw data without manual design. Deep learning models consist of a large number of spoofing data parameters that allow end-to-end learning, mapping from the original spoofing data input to the final detection of the spoofing output. This allows deep learning models to adapt to complex spoofing datasets and achieve better accuracy on spoofing detection tasks. Since deep learning models usually consist of multiple nonlinear layers, deep learning models can capture the nonlinear relationships in spoofing data. Deep learning models have a huge advantage over large-scale deceptive data because of their highly parallelizable structure. Among the selected deep learning models, MLP is only suitable for dealing with one-dimensional vector data and problems that do not have significant spatial structure. Therefore, MLP is not suitable for dealing with deceptive datasets with a large number of feature dimensions. CNNs can effectively learn the spatial structure in the input spoofing data through convolutional and pooling layers, and can gradually extract a high-level feature representation of the input spoofing data. In addition, they can reduce the number of parameters to be learned by sharing the parameters of convolutional kernels. Therefore, CNNs can effectively reduce the storage requirement

and computational complexity when dealing with spoofing datasets, thus improving the performance of the model. Both LSTM and GRU are used to process sequential data, and they have similar functions in dealing with long-term dependencies and memory capabilities. Therefore, they are suitable for processing deceptive data with sequential properties. LSTM has more complex mechanisms to control the flow of information, more sophisticated gating mechanisms, and more powerful modeling capabilities compared with GRU. Therefore, in dealing with spoofing data, LSTM is often better able to capture and utilize the information of features in spoofing data. The PCA-CNN-LSTM model has better computational performance in dealing with the problem of UAV GPS signal spoofing detection.



**Figure 16.** Accuracy (a), Precision (b), Recall (c) and F1\_Score (d) of 15 models on GPS signal spoofing dataset.



**Figure 17.** Confusion matrix of 15 spoofing models on GPS signal dataset. (a) PCA-Decision Tree; (b) PCA-KNN; (c) PCA-LR; (d) PCA-NB; (e) PCA-SVM; (f) PCA-Random Forest; (g) PCA-AdaBoost; (h) PCA-GBDT; (i) PCA-LightGBM; (j) PCA-XGBoost; (k) PCA-MLP; (l) PCA-LSTM; (m) PCA-GRU; (n) PCA-CNN-LSTM; (o) PCA-CNN-GRU.

## 6. Discussion and Conclusions

This paper proposes a PCA-CNN-LSTM model to detect GPS signal spoofing. Firstly, we used a composite-wing UAV and a deception spoofing jammer to collect a real dataset in a real scene. Next, we enhanced the real dataset using SVM-SMOTE. We analyzed the correlation between spoofing in the data features using the Spearman's rank correlation coefficient, and PCA dimensionality reduction was performed on the data features, where the number of features was reduced from 64 to 15. Then, we built a machine learning model for GPS signal spoofing detection. We used the grid search method to find the hyperparameters in the machine learning algorithm, and we adopted the 10-fold cross-validation method to train and test the model, which includes PCA-Decision Tree, PCA-KNN, PCA-LR, PCA-NB, PCA-SVM, PCA-Random Forest, PCA-AdaBoost, PCA-GBDT, PCA-LightGBM, and PCA-XGBoost. We also used early stopping to train the neural network model, which includes PCA-MLP, PCA-LSTM, PCA-GRU, PCA-CNN-LSTM, and PCA-CNN-GRU. We used confusion matrices as an evaluation method for computing the model, as well as a 10-fold cross-validation method to test the computational performance

of the model. The simulation results show that the PCA-CNN-LSTM model had the highest performance, with an accuracy of 99.43%.

In this study, we used a composite-wing UAV to collect real GPS signal data. Our GPS signal dataset has reliability and authenticity compared with the dataset generated by simulation and effectively retains the data features useful for GPS signal spoofing. Finally, we used the PCA-CNN-LSTM neural network algorithm to build a GPS signal spoofing detection model, which proves that the PCA-CNN-LSTM deep learning method has great potential in UAV spoofing detection. However, the route planned for the composite-wing UAV in our GPS signal dataset is only a combination of straight lines and arcs, and the complexity of path planning is not high. Moreover, a good neural network model usually has a large number of neurons, hidden layers, and features, which means that the computing requirements continue to increase, as does the storage space needed to hold the model and move data. These are difficult requirements for drones to meet. Therefore, in future studies, we plan to use more complex flight paths as GPS signal datasets, which can further improve the computational performance and generalization ability of the GPS signal spoofing detection model. In addition, we will use network pruning and quantitative perception training methods to carry out lightweight processing on the PCA-CNN-LSTM model. We will focus on reducing the complexity of the model while maintaining the computational performance of the model so that the GPS signal spoofing detection model can be deployed in the small UAV system. Moreover, the current research shows that UAV GPS signal spoofing detection studies have been selected for simpler deep learning fusion models, and we will propose new deep learning models and methods based on the feature information of spoofing data in later studies.

**Author Contributions:** Conceptualization, Y.S. and M.Y.; methodology, Y.S.; software, Y.S.; validation, Y.S., L.W. and T.L.; formal analysis, L.W.; investigation, T.L.; resources, M.D.; data curation, L.W.; writing—original draft preparation, Y.S.; writing—review and editing, M.Y.; visualization, T.L.; supervision, M.D.; project administration, M.D.; funding acquisition, M.D. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by General Project of Science and Technology Plan of Beijing Municipal Education Commission grant number KM202011232007, Programme of Introducing Talents of Discipline to Universities grant number D17021 and Connotation Development Project of Beijing Information Science and Technology grant number 2019KYNH204.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** Y.S. would like to thank Guangzhou Nansha Intelligent Photonic Sensing Research Institute for supporting this work and the funding from Key Laboratory of Photoelectric Testing Technology and Instrument, Ministry of Education.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Bao, L.; Wu, R.; Wang, W.; Lu, D. Spoofing mitigation in Global Positioning System based on C/A code self-coherence with array signal processing. *J. Commun. Technol. Electron.* **2017**, *62*, 66–73. [\[CrossRef\]](#)
2. Hartmann, K.; Giles, K. UAV exploitation: A new domain for cyber power. In Proceedings of the 2016 8th International Conference on Cyber Conflict (CyCon), Tallinn, Estonia, 31 May–3 June 2016; IEEE: Piscataway, NJ, USA, 2016.
3. Hartmann, K.; Steup, C. The vulnerability of UAVs to cyber attacks—An approach to the risk assessment. In Proceedings of the 2013 5th International Conference on Cyber Conflict (CYCON 2013), Tallinn, Estonia, 4–7 June 2013; IEEE: Piscataway, NJ, USA, 2013.
4. Liu, Y.; Li, S.; Fu, Q.; Liu, Z. Impact assessment of GNSS spoofing attacks on INS/GNSS integrated navigation system. *Sensors* **2018**, *18*, 1433. [\[CrossRef\]](#) [\[PubMed\]](#)
5. Psiaki, M.L.; Humphreys, T.E.; Stauffer, B. Attackers can spoof navigation signals without our knowledge. Here’s how to fight back GPS lies. *IEEE Spectr.* **2016**, *53*, 26–53. [\[CrossRef\]](#)
6. Psiaki, M.L.; Powell, S.P.; O’Hanlon, B.W. GNSS spoofing detection using high-frequency antenna motion and carrier-phase data. In Proceedings of the 26th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GNSS+ 2013), Nashville, TN, USA, 16–20 September 2013.

7. Ruckle, L.J. The Dragon Lady and the Beast of Kandahar: Bush and Obama-era US Aerial Drone Surveillance Policy Based on a Case Study Comparison of the 1960 U-2 Crash with the 2011 RQ-170 Crash. In *Technology and the Intelligence Community: Challenges and Advances for the 21st Century*; Springer: Cham, Switzerland, 2018; pp. 83–113.
8. Liang, C.; Miao, M.; Ma, J.; Yan, H.; Zhang, Q.; Li, X.; Li, T. Detection of GPS spoofing attack on unmanned aerial vehicle system. In *Proceedings of the Machine Learning for Cyber Security: Second International Conference, ML4CS 2019, Xi'an, China, 19–21 September 2019*; Springer: Cham, Switzerland, 2019.
9. Zhang, B.; Teunissen, P.J.G.; Yuan, Y.; Zhang, H.; Li, M. Joint estimation of vertical total electron content (VTEC) and satellite differential code biases (SDCBs) using low-cost receivers. *J. Geod.* **2018**, *92*, 401–413. [\[CrossRef\]](#)
10. Nasser, A.; Hassan, H.A.H.; Chaaya, J.A.; Mansour, A.; Yao, K.-C. Spectrum sensing for cognitive radio: Recent advances and future challenge. *Sensors* **2021**, *21*, 2408. [\[CrossRef\]](#) [\[PubMed\]](#)
11. Mohanti, S.; Soltani, N.; Sankhe, K.; Jaisinghani, D.; Di Felice, M.; Chowdhury, K. AirID: Injecting a custom RF fingerprint for enhanced UAV identification using deep learning. In *Proceedings of the GLOBECOM 2020—2020 IEEE Global Communications Conference, Taipei, Taiwan, 7–11 December 2020*; IEEE: Piscataway, NJ, USA, 2020.
12. Meng, L.; Yang, L.; Ren, S.; Tang, G.; Zhang, L.; Yang, F.; Yang, W. An approach of linear regression-based UAV GPS spoofing detection. *Wirel. Commun. Mobile Comput.* **2021**, *2021*, 5517500. [\[CrossRef\]](#)
13. Shafique, A.; Mehmood, A.; Elhadef, M. Detecting signal spoofing attack in uavs using machine learning models. *IEEE Access* **2021**, *9*, 93803–93815. [\[CrossRef\]](#)
14. Talaie Khoei, T.; Ismail, S.; Kaabouch, N. Dynamic selection techniques for detecting GPS spoofing attacks on UAVs. *Sensors* **2022**, *22*, 662. [\[CrossRef\]](#)
15. Wei, X.; Sun, C.; Lyu, M.; Song, Q.; Li, Y. ConstDet: Control Semantics-Based Detection for GPS Spoofing Attacks on UAVs. *Remote Sens.* **2022**, *14*, 5587. [\[CrossRef\]](#)
16. Nayfeh, M.; Li, Y.; Al Shamaileh, K.; Devabhaktuni, V.; Kaabouch, N. Machine Learning Modeling of GPS Features with Applications to UAV Location Spoofing Detection and Classification. *Comput. Secur.* **2023**, *126*, 103085. [\[CrossRef\]](#)
17. Feng, Z.; Guan, N.; Lv, M.; Liu, W.; Deng, Q.; Liu, X.; Yi, W. Efficient drone hijacking detection using two-step GA-XGBoost. *J. Syst. Arch.* **2020**, *103*, 101694. [\[CrossRef\]](#)
18. Wang, S.; Wang, J.; Su, C.; Ma, X. Intelligent detection algorithm against uavs' gps spoofing attack. In *Proceedings of the 2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS), Hong Kong, China, 2–4 December 2020*; IEEE: Piscataway, NJ, USA, 2020.
19. Shafiee, E.; Mosavi, M.R.; Moazedi, M. Detection of spoofing attack using machine learning based on multi-layer neural network in single-frequency GPS receivers. *J. Navig.* **2018**, *71*, 169–188. [\[CrossRef\]](#)
20. Jullian, O.; Otero, B.; Stojilović, M.; Costa, J.J.; Verdú, J.; Pajuelo, M.A. Deep Learning Detection of GPS Spoofing. In *Proceedings of the Machine Learning, Optimization, and Data Science: 7th International Conference, LOD 2021, Grasmere, UK, 4–8 October 2021; Revised Selected Papers, Part*. Springer: Cham, Switzerland, 2022.
21. Dang, Y.; Benzaid, C.; Yang, B.; Taleb, T.; Shen, Y. Deep-Ensemble-Learning-Based GPS Spoofing Detection for Cellular-Connected UAVs. *IEEE Internet Things J.* **2022**, *9*, 25068–25085. [\[CrossRef\]](#)
22. Sung, Y.-H.; Park, S.-J.; Kim, D.-Y.; Kim, S. GPS Spoofing Detection Method for Small UAVs Using 1D Convolution Neural Network. *Sensors* **2022**, *22*, 9412. [\[CrossRef\]](#)
23. Wu, S.; Li, Y.; Wang, Z.; Tan, Z.; Pan, Q. A Highly Interpretable Framework for Generic Low-Cost UAV Attack Detection. *IEEE Sens. J.* **2023**, *23*, 7288–7300. [\[CrossRef\]](#)
24. Kang, Q.; Shi, L.; Zhou, M.; Wang, X.; Wu, Q.; Wei, Z. A distance-based weighted undersampling scheme for support vector machines and its application to imbalanced classification. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *29*, 4152–4165. [\[CrossRef\]](#) [\[PubMed\]](#)
25. Napierala, K.; Stefanowski, J. Types of minority class examples and their influence on learning classifiers from imbalanced data. *J. Intell. Inf. Syst.* **2016**, *46*, 563–597. [\[CrossRef\]](#)
26. Kovács, G. An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets. *Appl. Soft Comput.* **2019**, *83*, 105662. [\[CrossRef\]](#)
27. Tao, X.; Li, Q.; Ren, C.; Guo, W.; Li, C.; He, Q.; Liu, R.; Zou, J. Real-value negative selection over-sampling for imbalanced data set learning. *Expert Syst. Appl.* **2019**, *129*, 118–134. [\[CrossRef\]](#)
28. Jolliffe, I.T.; Cadima, J. Principal component analysis: A review and recent developments. *Philos. Trans. R. Soc. Math. Phys. Eng. Sci.* **2016**, *374*, 20150202. [\[CrossRef\]](#)
29. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision, Washington, DC, USA, 7–13 December 2015*.
30. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.