WEP Warwick Evans Publishing

International Journal of Computer Science and Information Technology

ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 5, Number 3, Year 2025
DOI: https://doi.org/10.62051/ijcsit.v5n3.07
Journal homepage: https://wepub.org/index.php/IJCSIT/index

A SSDF Attack Detection Algorithm Based on Integration Learning and Deep Learning

Xiaojuan Bai, Jing Xu *, Shenghui Wang

College of Physics and Electronic Engineering, Northwest Normal University, Lanzhou, China
*Corresponding Author: xj1352024@nwnu.edu.cn

ABSTRACT

Spectrum sensing data falsification (SSDF) attacks are one of the most serious security threats in collaborative spectrum sensing models. In order to deal with SSDF attacks, an attack detection algorithm combining ensemble learning and deep learning is proposed. The algorithm uses soft voting mechanism to integrate the advantages of traditional integrated learning methods of support vector machine (SVM), decision tree and random forest, fully mining the features of the data, combined with the powerful automatic feature extraction capability of deep learning, to achieve accurate detection of SSDF attacks. At the same time, the multi-mode hybrid attack training and detection system can enhance the detection accuracy through continuous optimization, and improve the flexibility and reliability of the system in practical application. Experiments show that the proposed algorithm can effectively identify SSDF attack behavior, perform stably under different attack scenarios, and have strong robustness, which can effectively guarantee the security of spectrum sensing system.

KEYWORDS

SSDF; Spectrum sensing; Ensemble learning; Deep learning; Accuracy

1. INTRODUCTION

Collaborative Spectrum Sensing (CSS) [1], as the core technology enabling dynamic spectrum access in cognitive radio, significantly enhances the reliability of spectrum detection through multi-node data fusion. However, the openness resulting from its distributed nature exposes the system to severe threats from Spectrum Sensing Data Falsification (SSDF) attacks. SSDF attacks represent one of the major security threats targeting CSS in cognitive radio networks. By falsifying or tampering with sensing data, attackers can interfere with the global decision-making of the Fusion Center (FC), leading to the waste of spectrum resources and even network paralysis.

With the widespread application of Cognitive Radio (CR), defense techniques against SSDF attacks have become a research hotspot. Traditional SSDF defense schemes rely on methods such as reputation evaluation and statistical hypothesis testing, but they exhibit significant limitations in dynamic attack environments: Detection based on Bayesian reasoning requires preset priors for attack distributions [2], with actual false detection rates as high as 25%; algorithms based on behavioral clustering have recognition rates of less than 50% for collaborative camouflage attacks [3]; and shallow machine learning models (such as SVM) generally have detection accuracy rates below 80% due to limited feature representation capabilities [4].

In recent years, deep learning technology has made breakthrough progress in areas such as image recognition and natural language processing due to its powerful feature extraction and nonlinear

modeling capabilities [5]. Introducing deep learning into the field of spectrum sensing, by constructing end-to-end signal classification models, can effectively uncover the intrinsic features and spatiotemporal correlations of signals under complex channel environments, significantly enhancing sensing accuracy and anti-interference capability. Research on integrating efficient spectrum sensing algorithms with deep learning and designing defense mechanisms against SSDF attacks is key to achieving secure and reliable dynamic spectrum sharing.

This paper establishes a multi-pattern attack model, including random attacks, global attacks, selective attacks, and intelligent attacks. Among these, the intelligent attack mode selects significant spectral features for data falsification and can adaptively adjust the attack intensity based on historical attack records. On this basis, an attack detection algorithm combining ensemble learning and deep learning is proposed. This method uses multiple attack strategies to train the detection model and tests its ability to detect attacking users. The functionality of each module within the system is detailed, and performance metrics for measuring the effectiveness of attacks are analyzed. Multiple comparative experiments verify that the proposed algorithm performs well across different attack scenarios, demonstrating its robustness and effectiveness.

2. SYSTEM MODEL

This paper addresses the issue of SSDF attack detection in centralized cooperative spectrum sensing systems, as shown in Fig. 1, includes a Fusion Center, one Primary User (PU), n Secondary Users (SUs), and m several Malicious Secondary Users (MSUs). The interference caused by malicious attacks on the system remains uncertain and unknown; therefore, $m = \beta \cdot n$, here β represents the proportion of malicious users among the SUs.

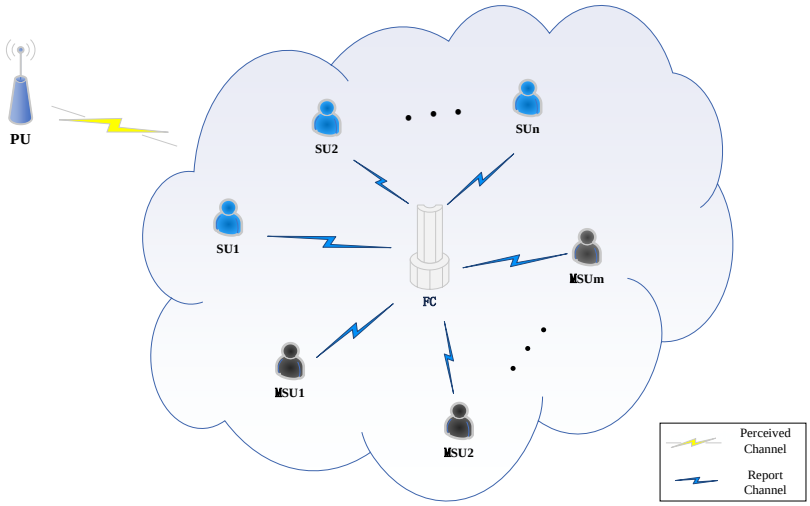


Figure 1. Spectrum sensing system model under SSDF attack

In a normal sensing system, each SU independently monitors and senses the PU signal and reports the sensed data results to the FC. The FC then makes a final decision on whether the PU signal truly exists in the channel based on all the aggregated data. Only when the FC reports that the PU spectrum is idle can the SUs access the PU channel for spectrum usage; otherwise, they must refrain from accessing to avoid interfering with the PU's normal operation.

In an attacked system, malicious users may exist with a certain probability, sending falsified and tampered data to the FC, thereby disrupting its decision-making process. This leads to situations where SUs are unable to find the correct timing to access the spectrum. To focus on addressing SSDF attacks and detection within the system, it is assumed that there is no interference among SUs, meaning they operate independently of one another. This assumption simplifies the analysis and design of effective defense mechanisms against SSDF attacks.

Modeling the spectrum sensing problem as a binary hypothesis problem, the signal received by the i^{th} SU at time instant n can be expressed as:

$$x_i(n) = \begin{cases} u_i(n), & H_0 \\ s(n) + u_i(n), & H_1 \end{cases} \tag{1}$$

Here, H_0 represents the state where the PU signal does not exist, and H_1 represents the state where the PU signal exists. $s(n)$ denotes the PU signal, and $u_i(n)$ denoted as represents the noise at that moment for the i^{th} SU, which is composed of independent identically distributed (i.i.d.) Gaussian white noise with a mean of 0 and a variance of σ^{2n}.

Each SU first performs local sensing and makes individual decisions. The detection probability P_d and false alarm probability P_f for the i^{th} SU regarding the PU signal can be expressed as follows:

$$P_d^{\,i} = P\{r_i = 1 \mid H_1\} \tag{2}$$

$$P_f^{\,i} = P\{r_i = 1 \mid H_0\} \tag{3}$$

In Eq. 2 and Eq. 3, r_i refers to the decision information provided by the i^{th} SU to the FC. When $r_i = 1$, it indicates that the PU is currently occupying the spectrum at that moment. Conversely, if $r_i = 0$, it indicates that the PU spectrum is idle. However, the information received by the FC cannot be guaranteed to be error-free during transmission. Malicious users may probabilistically tamper with the data, leading to incorrect overall decisions by the FC.

3. ATTACK MODEL

One of the research focuses of this paper is to simulate complex SSDF attacks, aiming to prevent detection algorithms that perform well against simple attacks from experiencing degraded performance or even failure when applied in complex communication environments. Therefore, the malicious users considered in this study do not merely attack the decision results of a single SU by flipping “0” to “1” or “1” to “0”. Instead, they directly attack the PU signals received by the SU at varying degrees, thereby increasing the difficulty of detection. The attack model is illustrated in Fig. 2. We propose four mechanisms for data tampering with PU signal characteristics, including random attack, global attack, selective attack, and intelligent attack. The attacker randomly selects one of these mechanisms when generating the characteristics of malicious users. This approach provides sample data for different attack modes, enabling the detector to learn various attack data characteristics during training and enhancing the model’s generalization ability. Consequently, the system can more effectively identify potential attackers.

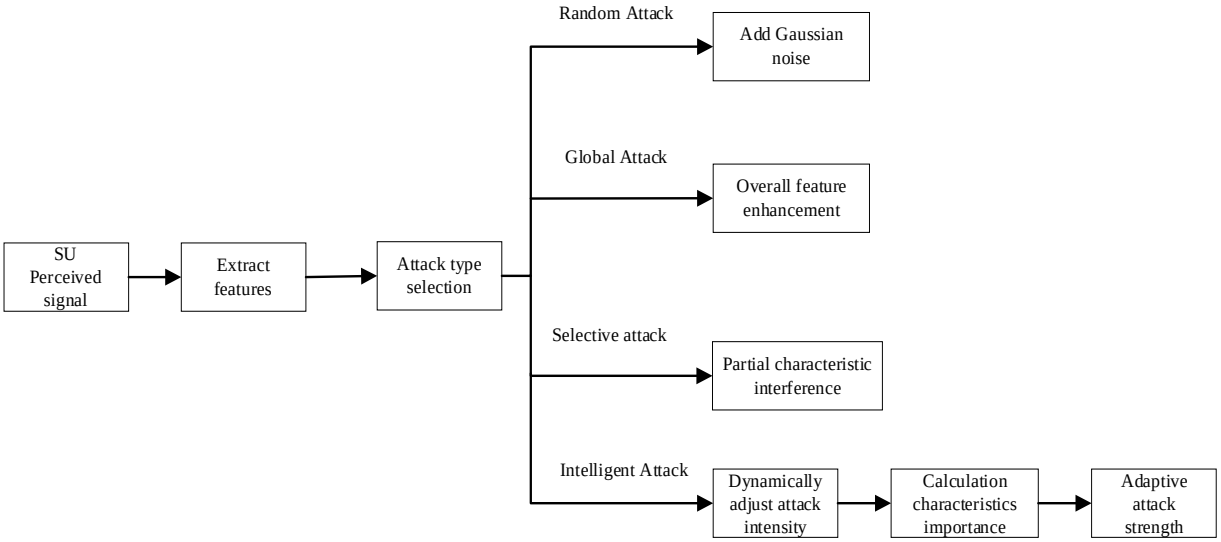


Figure 2. Attack model

The detection system selects the energy feature, power feature, and cyclostationary feature of the signal for extraction as the dataset for the model, denoted as

$$E = \sum_{n=0}^{N-1} | x(n) |^2, n = 0,1,\cdots, N-1 \tag{4}$$

$$P = \frac{1}{NT_s} \sum_{n=0}^{N-1} | x(n) |^2 \tag{5}$$

$$R = \frac{1}{N} \sum_{n=0}^{N-1} x(n) x^*(n+k) e^{-j2\pi\alpha n} \tag{6}$$

In Eq. 5, T_s represents the sampling period, and in Eq. 6, α represents the cyclostationary frequency.

The original features that have not been attacked are combined into a feature vector $\mathbf{x}=[E,P,R]=[x_1,x_2,x_3]$, forming dataset $X=\{\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_n\}$. Let's denote all feature vectors that have suffered from SSDF attacks collectively as $\mathbf{x}'=[x_1',x_2',x_3']$. Standardization is performed on all features, calculated as follows:

$$z_i = \frac{x_i - \mu_i}{\sigma_i}, i = 1,2,3 \tag{7}$$

Where, $\mu_i = \frac{1}{N} \sum_{n=1}^{N} x_i^{(n)}$ represents the mean of the feature, and $\sigma_i = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} \left(x_i^{(n)} - \mu_i \right)^2}$ represents the standard deviation of the feature.

Random Attack: MSU tampers with data by non-uniformly interfering with all features, thereby disrupting the inherent correlations between them. A direct and easily implementable approach is to use a random number attack, where we add independent Gaussian noise to all selected features under attack. The feature vector after the attack can be represented as

$$\mathbf{x}' = \mathbf{x} + \varepsilon, \varepsilon \sim \mathrm{N}(0, \sigma^2) \tag{8}$$

Where $\mathbf{x}'$ represents the signal features after being attacked by a malicious user, $\mathbf{x}$ represents the original features, σ represents the attack intensity, and ε denotes noise that follows a Gaussian distribution. After standardization, the corresponding z' can be obtained as

$$z_i' = \frac{x_i + \varepsilon_i - \mu_i}{\sigma_i} = z_i + \frac{\varepsilon_i}{\sigma_i}, i = 1, 2, 3 \tag{9}$$

Global Attack: MSU alters the magnitude of all features as a whole, either amplifying or reducing all features by a constant proportion. The attack outcome is as follows:

$$\mathbf{x}' = \mathbf{x} \cdot (1 + \alpha) \tag{10}$$

Where α represents the attack intensity, and the standardized features are:

$$z_i' = \frac{(1+\alpha)x_i - \mu_i}{\sigma_i}, i = 1, 2, 3 \tag{11}$$

Selective Attack: MSU randomly selects a subset of features with a certain probability to attack, while keeping some features normal in order to deceive detection models.

Intelligent Attack: The attacker selectively tampers with features based on their importance, simulating advanced malicious users. This increases the stealth and targeting of the attack, making it more realistic in complex communication environments. We assume that the larger the absolute value of a feature, the more important it is. First, features are extracted and their importance is calculated to generate an attack mask. This mask is used to determine which features have absolute values greater than the median of all feature absolute values. The attacker then selects features to attack based on this mask. The calculation of the mask is as follows:

$$M_j = \begin{cases} 1, & |\mathrm{x}| > median(\mathrm{x}) \\ 0, & \text{else} \end{cases} \tag{12}$$

If the result of the mask is "1", then the j^{th} important feature is selected as the target for tampering, and the feature vector after the attack is

$$\mathbf{x}' = \mathbf{x} \cdot \left(1 + \alpha \cdot rand()\right) \tag{13}$$

In the equation, α represents the attack intensity. After standardizing the features under intelligent attacks, we obtain:

$$z_j' = \frac{x_j(1 + \alpha \cdot rand()) - \mu_j}{\sigma_j}, j \subseteq \{1, 2, 3\} \tag{14}$$

By establishing a feedback mechanism to detect the attack effect, the results of whether each attack is accurately identified by the detector are saved as historical records. By observing the success rate of historical attacks, the attacker adaptively adjusts the attack intensity. When the success rate is low, the attack intensity is increased to enhance the attack effect; when the success rate is high, the attack intensity is appropriately reduced to avoid detection of the attack. The attack intensity is continuously

and dynamically optimized to make the attack more concealed and effective. The method for adjusting the attack intensity is as follows:

$$\alpha_{t+1} = \alpha_t \cdot \exp(\eta \cdot (r_t - \tau)) \tag{15}$$

Where, α_t represents the intensity of the previous attack, r_t represents the success rate of the previous attack, τ is the set threshold for the target success rate, and η is the learning rate.

As the detection system undergoes continuous training and optimization, its detection capabilities will also change. Adjusting the attack intensity allows the attack strategy to adapt to changes in the detection system, thereby increasing the likelihood of a successful attack. The fusion of different attack modes can make attack behaviors more diversified and unpredictable, increasing the difficulty for the detection system to accurately identify attacking users. Under the pressure of various attacks, the detection system needs to continuously optimize its parameters through training to effectively identify attacking users. This can improve the flexibility and reliability of the detection system in practical applications.

4. DESIGN OF DETECTION ALGORITHM COMBINING ENSEMBLE LEARNING AND DEEP LEARNING

The detection network adopts a combination of ensemble learning models and deep learning models. Ensemble learning includes SVM, Random Forest, and Decision Trees, as shown in Fig. 3. Ultimately, a soft voting mechanism is used to fuse the prediction results of different models, selecting the prediction with higher confidence to improve the accuracy of the final prediction.

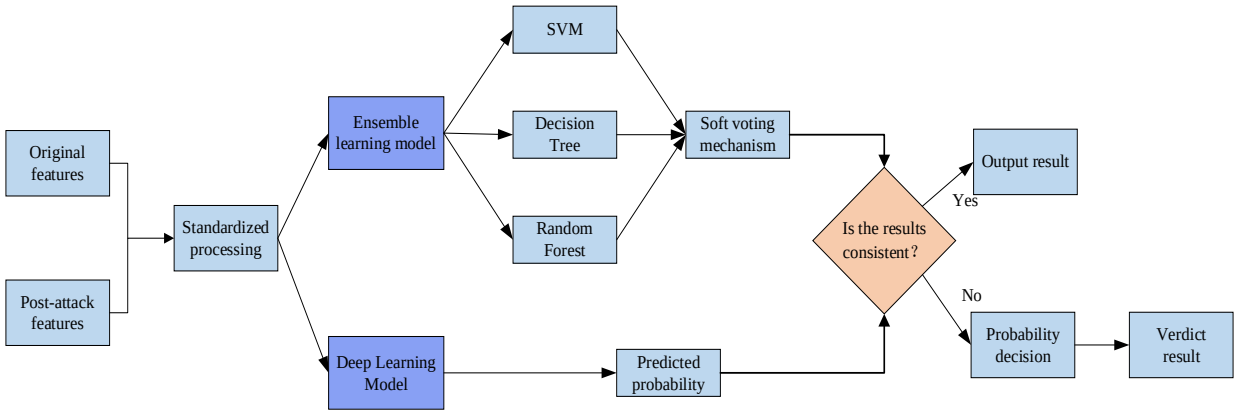


Figure 3. Design of detection algorithm combining ensemble learning and deep learning

After standardizing the signal features of both the non-attacked SU and the attacked MSU, their respective feature vectors $\mathbf{z} = [z_1, z_2, z_3]$ and $\mathbf{z}' = [z_1', z_2', z_3']$ are obtained. Then, both ensemble learning models and deep learning models are used to train and classify the feature data. If the prediction results of the two models are consistent, the prediction result is directly output to determine whether the current signal is under SSDF attack. If the prediction results of the two models conflict, the prediction result with higher confidence is selected as the final classification result, which is referred to as "probability decision" in Fig. 3. The prediction result of the ensemble learning model is a probability matrix [$P_{ENS}(0|x)$, $P_{ENS}(1|x)$], where $P_{ENS}(0|x)$ and $P_{ENS}(1|x)$ represent the probabilities of predicting the class as the negative sample "0" and the positive sample "1" respectively. The output of the deep learning model only represents the probability $P_{DL}(1|x)$ of predicting the class as the positive sample "1". When using probability decision-making, the combined probability of predicting the positive sample by the two models is calculated as

$$P(1|x)=\frac{P_{ENS}(1|x)+P_{DL}(1|x)}{2} \tag{16}$$

Therefore, the probability of predicting a negative class sample is

$$P(0|x)=1-P(1|x) \tag{17}$$

From this, a confidence matrix $\begin{bmatrix} P(0|x) \\ P(1|x) \end{bmatrix}$ is obtained. By comparing the probabilities in the matrix, the final predicted classification result can be determined.

4.1. Integrated Learning Classification Module

Ensemble learning consists of SVM, random forest (RF) and decision tree (DT), which are fused by soft voting, and the weight of the three mechanisms is set to 2:1:1.

(1) SVM

Support Vector Machine (SVM) is a supervised learning model for binary classification capable of performing tasks such as classification and regression. Its main idea is to find a hyperplane that maximizes the margin between different classes [6]. In the context of SSDF detection tasks, for a given training dataset $\{(x_i,y_i)\},i=1,\cdots,m$, where x_i represents the feature vectors and $y_i\in\{-1,+1\}$ represents the label categories, a value of "-1" indicates a negative instance (normal SU data), while "+1" indicates a positive instance (MSU data). The objective of SVM is to find a hyperplane $w\cdot x+b=0$ based on the extracted features, which separates normal data from data attacked by SSDF. Here, $w=(w_1,w_2,\cdots,w_m)$ represents the normal vector of the hyperplane, $x=(x_1,x_2,\cdots,x_m)$ is the input feature vector, and b is the intercept. SVM determines the parameters of the hyperplane by maximizing the minimum distance or margin between the two classes of feature data to the hyperplane. The margin can be denoted as $\gamma=\frac{2}{\|w\|}$, where $\|w\|=\sqrt{\sum_{i=1}^{m}w_i^2}$, then the optimization problem [7] can be formulated as

$$\min_{w,b}\frac{1}{2}\|w\|^2, s.t.y_i(w\cdot x_i+b)..1,i=1,2,\cdots,n \tag{18}$$

Due to the fact that the signal features are non-linearly separable data, we adopt the Gaussian kernel function (Radial Basis Function, RBF) to map the feature data into a high-dimensional space for classification [8]. The Gaussian kernel function is expressed as

$$K(x_i,x_j)=\exp(-\gamma\cdot\|x_i-x_j\|^2),\gamma>0 \tag{19}$$

Where, the parameter γ controls the scope of the kernel function, which is also known as the bandwidth. When the value of γ is large, the similarity between two samples decreases rapidly as their distance increases [9]. $K(x_i,x_j)$ represents the kernel function value between the i^{th} and j^{th} feature vectors, measuring their similarity in the high-dimensional feature space. A value closer to 1 indicates that the features are more similar, while a value closer to 0 suggests a greater difference.

In SVM, besides the kernel function, the regularization parameter [10] C also plays a significant role, representing the model's tolerance for errors in the training data. When the value of C is large, it can reduce classification errors but may also lead to increased model complexity. Conversely, when

the value of C is small, the model tends to find a smoother and simpler decision boundary, which means it may accept more classification errors. Therefore, selecting an appropriate value of C based on the requirements of the system and the dataset is crucial.

In the network structure designed in this paper, the regularization parameter $C=1.0$, and the value of γ is adjusted automatically. Using the kernel matrix computed from the feature vectors x and the corresponding labels, the SVM model is trained to find the optimal classification hyperplane. Then, the trained SVM model is utilized to determine whether the input features belong to malicious users.

(2) Decision Tree

A decision tree constructs a tree-like structure for classifying samples by recursively splitting features [11]. Each internal node in a decision tree represents a test on an attribute, each branch represents an outcome of that test, and each leaf node represents a class. The Gini index is selected as the criterion for splitting to choose the optimal feature for division, thereby constructing an effective classification model.

Assuming D is a dataset and D_k is the subset of samples belonging to the kth class in the dataset, if the proportional representation of this class $p_k = \frac{D_k}{D}$, the Gini index [12] can be calculated as follows:

$$Gini(D) = 1 - \sum_{k=1}^{K} {p_k}^2 \tag{20}$$

Where K=2, representing the two classes of “normal” and “attack”.

When splitting at each node, for a certain value a of feature A, the Gini index is calculated as follows:

$$Gini(A = a) = \sum_{k=1}^{K} \frac{|D_k|}{|D|} Gini(D_k), K = 2 \tag{21}$$

In the formula, $|D|$ and $|D_k|$ represent the number of samples in the sets D and D_k, respectively. The feature and its value that minimize the Gini index are selected as the optimal split for the current node. When constructing the decision tree, the feature with the smallest $Gini(A=a)$ is chosen as the splitting feature for the current node, and the tree structure is recursively built until certain conditions are met. The maximum depth of the decision tree is set to 8, and the minimum number of samples required for a node to split is set to 5.

(3) Random Forest

Random forest improves the accuracy and stability of classification by constructing multiple decision trees and averaging or voting their prediction results. Each tree is generated by sampling with replacement from the original dataset to create a training subset, and only a subset of features is randomly considered when selecting split nodes [13]. Similar to decision trees, the Gini index is used as the splitting criterion. However, the difference is that the selection of feature subsets in random forest is random, which can reduce overfitting and improve the generalization ability of the model. When building the model, the number of trees is selected to be 200, the maximum depth is set to 8, the minimum number of samples required for a node to split is set to 5, and the final classification result is determined by the voting of all trees.

The ensemble strategy for the three models adopts a soft voting mechanism. Assuming the prediction probabilities of SVM, decision tree, and random forest are $P_{SVM} = (p_1^{SVM}, p_{-1}^{SVM})$, $P_{DT} = (p_1^{DT}, p_{-1}^{DT})$, and $P_{RF} = (p_1^{RF}, p_{-1}^{RF})$, respectively, where p_1 represents the probability of predicting normal signal characteristics and p_{-1} represents the probability of predicting attacked signal characteristics. According to the soft voting mechanism with a ratio of 2:1:1, the final prediction probability $P = (p_1, p_{-1})$ is calculated as follows:

$$p_1 = \frac{2p_1^{SVM} + p_1^{DT} + p_1^{RF}}{4} \tag{22}$$

$$p_{-1} = \frac{2p_{-1}^{SVM} + p_{-1}^{DT} + p_{-1}^{RF}}{4} \tag{23}$$

If the result is $p_1 > p_{-1}$, then the predicted signal characteristics are judged to be from a normal SU; otherwise, they are judged to be from an MSU after an SSDF attack.

The ensemble learning model combines the advantages of SVM, random forest, and decision tree models. It not only enhances the overall performance of the model but also determines the final classification result based on the prediction probabilities of each model, further improving classification accuracy.

4.2. Deep Learning Prediction Module

The complete detection framework also includes a deep learning model to complement the shortcomings of the ensemble learning model. By combining the prediction results of the ensemble learning model and the deep learning model, the detection of SSDF attacks can achieve better robustness. The deep learning network learns more complex relationships among features from the input features through multiple layers of nonlinear transformations, extracts high-order abstract representations of features, and ultimately achieves binary classification.

The network structure consists of an input layer, an output layer, and two hidden layers. For the input feature $x = (x_1, x_2, \cdots, x_n)$, the input to the hidden layer is denoted as $z_j = \sum_{i=1}^{n} w_{ij}^1 x_i + b_j^1$, where w_{ij}^1 represents the weights from the input layer to the hidden layer, and b_j^1 represents the bias of the hidden layer. The output of the hidden layer is denoted as $a_j = f(z_j)$, where $f(\cdot)$ is the ReLU activation function used in the hidden layer, $f(x) = \max(0, x)$. The output layer employs the Softmax activation function $f(x) = \frac{1}{1+e^{-x}}$ to output the probability of an attack.

During the model training process, the Adam optimizer is selected with a learning rate set to 0.001, and binary cross-entropy loss is used as the loss function[14]. The calculation process is as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(p_i) + (1-y_i)\log(1-p_i)\right] \tag{24}$$

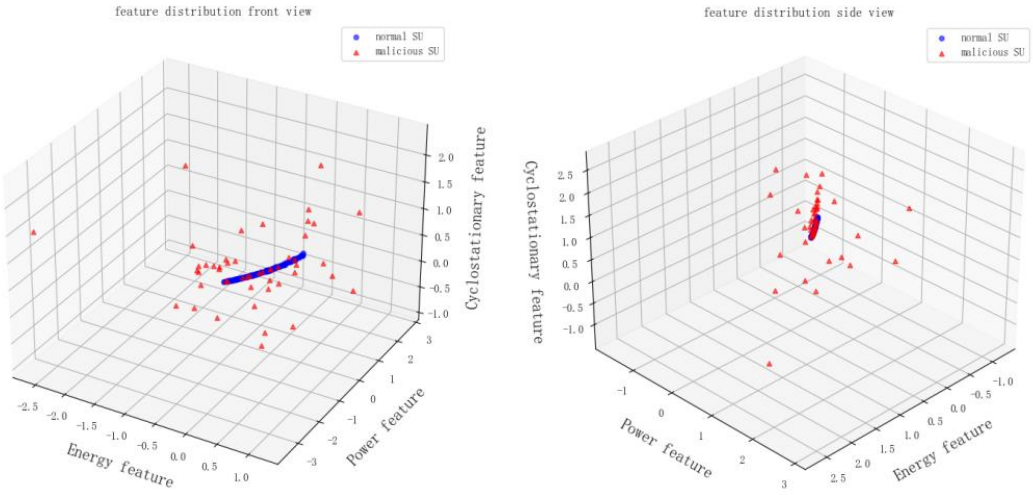
Where, N denotes the total number of samples in the dataset, y_i denotes the true label (0 or 1) of the i^{th} sample, and p_i denotes the predicted attack probability by the model. During each iteration, the model is trained for one epoch, and its performance is evaluated on the validation set. The performance metrics for each epoch are recorded.

In summary, the deep learning model in the SSDF attack detection system is primarily responsible for learning deep feature representations of the input feature data. By combining with the ensemble model, it achieves effective detection of SSDF attacks. This combination not only fully leverages the powerful feature learning capability of deep learning but also retains the efficiency of traditional ensemble learning models on specific tasks.

5. EXPERIMENT AND PERFORMANCE ANALYSIS

During the experiment, QPSK modulated signals are used to simulate PU signal data, and energy features, power features, and cyclostationary features are extracted to form the dataset for model training. The training set and test set are split in an 8:2 ratio. The number of SUs is set to 200, and the number of malicious users is 50.

Fig. 4 is a 3D scatter plot of the signal feature distribution, where blue dots represent the features of normal SUs, and red triangles represent the features of attacked MSUs. The classification of the two types of data can be visually observed. From the figure, it can be seen that the features of the unattacked SUs are concentrated and exhibit a certain regularity, while the features of the attacked MSUs are scattered and irregular.



(a) Front view (b) Side view

Figure 4. 3D feature distribution of normal SU and malicious SU

When studying the SSDF detection algorithm that combines ensemble learning and deep learning, besides detection probability and false alarm probability, accuracy, precision, recall, and F1-score are also important indicators for evaluating the performance of the algorithm [15]. They can more comprehensively measure the different performances of the algorithm in applications.

(1) Accuracy refers to the proportion of correctly predicted samples by the model to the total number of samples, comprehensively evaluating the overall classification accuracy of the algorithm. The calculation method is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{25}$$

Where, TP represents the number of samples correctly predicted as attacks, known as True Positives; TN represents the number of samples correctly predicted as normal, known as True Negatives; FP represents the number of normal samples falsely predicted as attacks, known as False Positives; and FN represents the number of attack samples falsely predicted as normal, known as False Negatives.

(2) Precision indicates the proportion of samples predicted as attacks that are actually attacks, reflecting the accuracy of the algorithm in identifying malicious users. The expression is

$$Precision = \frac{TP}{TP + FP} \tag{26}$$

(3) Recall refers to the proportion of actual attack samples that are correctly detected, reflecting the model's ability to identify attacked signals. The calculation formula is

$$Recall = \frac{TP}{TP + FN} \tag{27}$$

(4) The F1-score is a performance metric that considers both precision and recall comprehensively, reflecting the balance between precision and recall in the algorithm. It is suitable for scenarios with class imbalance. The expression is

$$F1score = 2 \cdot \frac{Precision \times Recall}{Precision + Recall} \tag{28}$$

(5) Robustness describes the ability of an algorithm to maintain its performance when faced with different types of attacks or environmental changes, such as SNR.

When the detection system has 200 normal SUs, 50 MSUs, with an attack intensity set to 0.3 and SNR=-10dB, the algorithm model is trained with 50 epochs, and the success rate threshold for intelligent attack modes is set to 0.7. The classification results of the model are visualized, and the confusion matrix is shown in Fig. 5. The classification results indicate that among the 50 samples in the test set, 40 normal SUs and 9 MSUs are correctly classified, with 1 normal SU being misclassified as an MSU, and no MSU samples being missed. This demonstrates good detection performance.

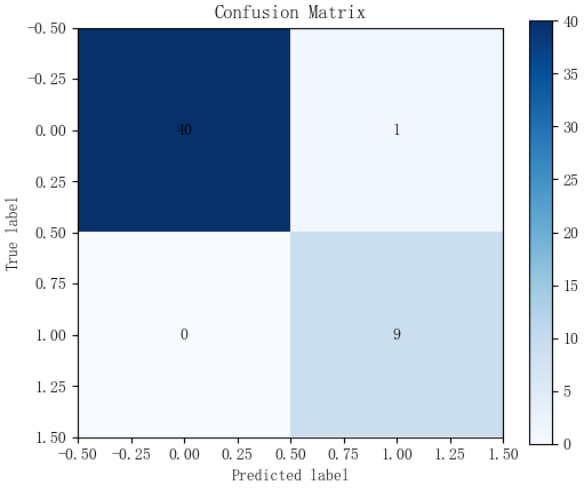


Figure 5. The confusion matrix of SSDF attack detection performance of the proposed algorithm

Fig. 6 presents the ROC curves of different detection models under SSDF attacks, illustrating the relationship between detection probability and various false alarm probabilities [16]. For SSDF detection systems, the goal is to achieve as high a detection probability as possible while keeping the false alarm probability low, ensuring the interception of multiple MSUs while reducing misjudgments of normal SUs. From the figure, it can be seen that the ROC curve of the algorithm proposed in this chapter leans more towards the top-left corner compared to other algorithms, indicating that this

algorithm achieves a higher detection probability while maintaining a low false alarm probability. When the false alarm probability is 0.42, the detection probability of the detection algorithm combining ensemble learning and deep learning reaches 1, while the P_d of other algorithms are still below 0.8. The overall AUC value is improved by approximately 0.07 to 0.28, demonstrating that this algorithm has superior attack detection performance.

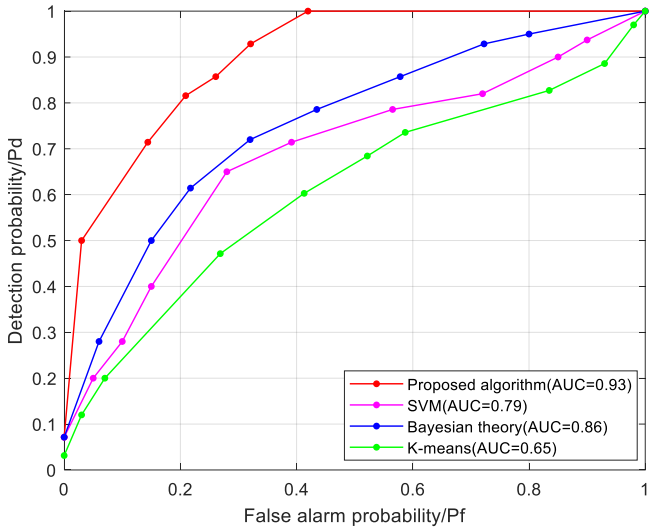
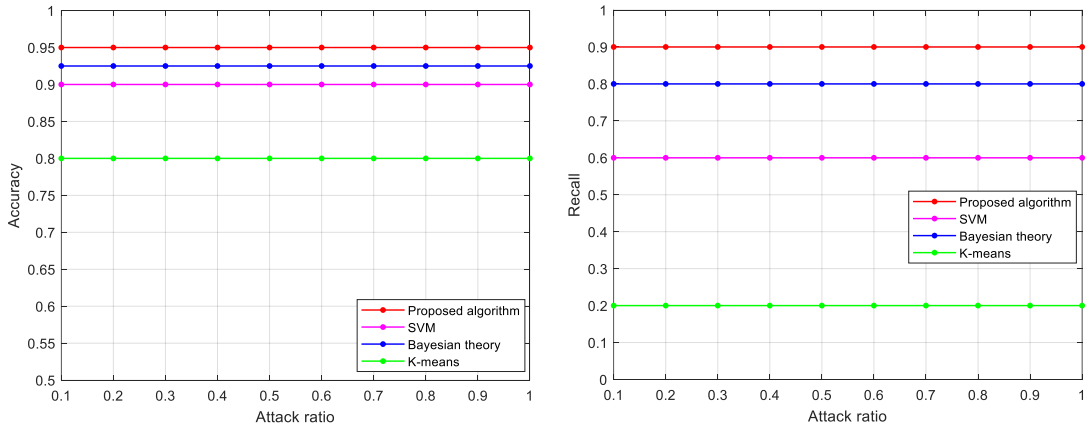


Figure 6. ROC curves of different SSDF attack detection algorithms

Fig. 7 displays the detection performance of different detection algorithms as the proportion of attacking users varies. Subfigure (a) compares the accuracy, and subfigure (b) compares the recall. The detection algorithm proposed in this chapter demonstrates significant advantages in performance and maintains stable detection results across different attack proportions, indicating its flexibility in practical applications.



(a) Accuracy (b) Recall

Figure 7. Algorithm performance comparison diagram with different attacker ratio

Fig. 8 presents the detection accuracy of the algorithm for attacking users in different SNR environments, with the SNR set from -20dB to 10dB in steps of 5dB. As can be seen from the figure, compared to other detection algorithms, the algorithm proposed in this chapter exhibits stronger detection capability at low SNR. In high SNR scenarios with less interference, the detection accuracy basically reaches 100%. The proposed algorithm maintains good stability under different noise conditions, demonstrating its robustness and adaptability.

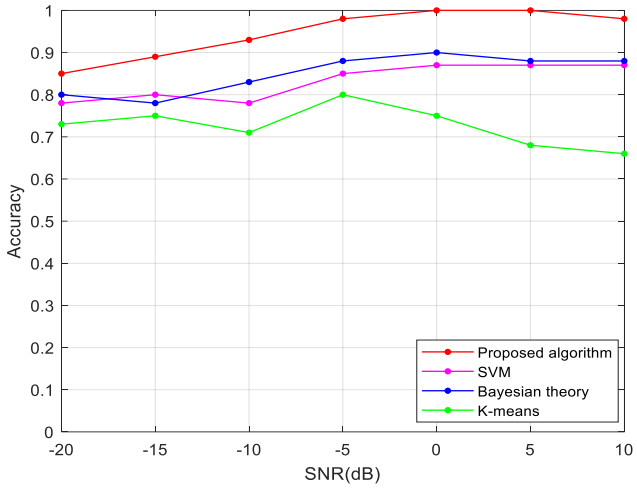


Figure 8. Comparison of detection accuracy of algorithms under different SNR

In summary, the SSDF detection algorithm combining ensemble learning and deep learning proposed in this chapter can effectively identify attacking users. Compared to traditional methods, it has higher accuracy and recall rates, as well as high robustness, making it stable for application in different communication environments and attack scenarios.

6. CONCLUSION

This paper addresses the SSDF attack problem in spectrum sensing by proposing an anti-SSDF attack detection algorithm that combines ensemble learning and deep learning. The ensemble learning module integrates the classification advantages of SVM, Decision Tree, and Random Forest through a soft voting mechanism, combined with the automatic extraction capability of deep neural networks for high-dimensional nonlinear features, to construct a multi-mode hybrid attack detection model. In the attack modeling, four types of attack modes are designed: random attack, global attack, selective attack, and adaptive intelligent attack. Among them, intelligent attacks dynamically adjust tampering strategies to simulate the evolutionary behavior of real attackers. This algorithm leverages the advantages of multiple classifier fusion in ensemble learning and the deep mining of attack features by deep learning models, enabling the algorithm to achieve high accuracy and low false alarm rates in detecting SSDF attacks. Experimental results validate the effectiveness of the algorithm in ensuring the security of spectrum sensing, capable of detecting SSDF attacks timely and accurately, providing strong support for the safe operation of spectrum sensing systems.

REFERENCES

[1] Akyildiz I F, Lo B F, Balakrishnan R. Cooperative spectrum sensing in cognitive radio networks: A survey [J]. Physical communication, 2011, 4(1): 40-62. https://doi.org/10.1016/j.phycom.2010.12.003

[2] Kaligineedi P, et al. Bayesian detection of malicious users in collaborative spectrum sensing [J]. IEEE Transactions on Wireless Communications, 2012. https://doi.org/10.1109/ACCESS.2017.2756992

[3] Li Y, Peng Q. Achieving secure spectrum sensing in presence of malicious attacks utilizing unsupervised machine learning [C]//MILCOM 2016-2016 IEEE Military Communications Conference. IEEE, 2016: 174-179. https://doi.org/10.1109/MILCOM.2016.7795321

[4] Sarmah R, Taggu A, Marchang N. Detecting Byzantine attack in cognitive radio networks using machine learning [J]. Wireless Networks, 2020, 26(8): 5939-5950. https://doi.org/10.1007/s11276-020-02398-w

[5] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition [J]. arXiv preprint arXiv:1409.1556, 2014. https://doi.org/10.48550/arXiv.1409.1556

[6] Cortes C, Vapnik V. Support-vector networks [J]. Machine learning, 1995, 20: 273-297. https://doi.org/10.1007/BF00994018

[7] Hastie T, Tibshirani R, Friedman J H, et al. The elements of statistical learning: data mining, inference, and prediction [M]. New York: springer, 2009. https://doi.org/10.1007/978-0-387-21606-5

[8] Omer G, Mutanga O, Abdel-Rahman E M, et al. Performance of support vector machines and artificial neural network for mapping endangered tree species using WorldView-2 data in Dukuduku forest, South Africa [J]. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 2015, 8(10): 4825-4840. https://doi.org/10.1109/JSTARS.2015.2461136

[9] Keerthi S S, Lin C J. Asymptotic behaviors of support vector machines with Gaussian kernel [J]. Neural computation, 2003, 15(7): 1667-1689. https://doi.org/10.1162/089976603321891855

[10] Wang S, Zhang J, Fu Y, et al. ACM transactions on intelligent systems and technology [C]//ACM Transactions on Intelligent Systems and Technology. 2011. https://doi.org/10.1145/1961189.1961199

[11] Quinlan J R. Induction of decision trees [J]. Machine learning, 1986, 1: 81-106. https://doi.org/10.1007/BF00116251

[12] Loh W Y. Classification and regression trees [J]. Wiley interdisciplinary reviews: data mining and knowledge discovery, 2011, 1(1): 14-23. https://doi.org/10.1002/widm.8

[13] Breiman L. Random forests [J]. Machine learning, 2001, 45: 5-32. https://doi.org/10.1023/A:1010933404324

[14] LeCun Y, Bengio Y, Hinton G. Deep learning [J]. nature, 2015, 521(7553): 436-444. https://doi.org/10.1038/nature14539

[15] Sokolova M, Lapalme G. A systematic analysis of performance measures for classification tasks [J]. Information processing & management, 2009, 45(4): 427-437. https://doi.org/10.1016/j.ipm.2009.03.002

[16] Fawcett T. An introduction to ROC analysis [J]. Pattern recognition letters, 2006, 27(8): 861-874. https://doi.org/10.1016/j.patrec.2005.10.010