remote sensing

MDPI

Article

Fast Complex-Valued CNN for Radar Jamming Signal Recognition

Haoyu Zhang, Lei Yu, Yushi Chen * and Yinsheng Wei

School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150001, China; 19S005093@stu.hit.edu.cn (H.Z.); yu.lei@hit.edu.cn (L.Y.); weiys@hit.edu.cn (Y.W.)
* Correspondence: chenyushi@hit.edu.cn

Abstract: Jamming is a big threat to the survival of a radar system. Therefore, the recognition of radar jamming signal type is a part of radar countermeasure. Recently, convolutional neural networks (CNNs) have shown their effectiveness in radar signal processing, including jamming signal recognition. However, most of existing CNN methods do not regard radar jamming as a complex value signal. In this study, a complex-valued CNN (CV-CNN) is investigated to fully explore the inherent characteristics of a radar jamming signal, and we find that we can obtain better recognition accuracy using this method compared with a real-valued CNN (RV-CNN). CV-CNNs contain more parameters, which need more inference time. To reduce the parameter redundancy and speed up the recognition time, a fast CV-CNN (F-CV-CNN), which is based on pruning, is proposed for radar jamming signal fast recognition. The experimental results show that the CV-CNN and F-CV-CNN methods obtain good recognition performance in terms of accuracy and speed. The proposed methods open a new window for future research, which shows a huge potential of CV-CNN-based methods for radar signal processing.

Keywords: radar jamming signal; recognition; convolutional neural network (CNN); model pruning; complex-valued network

Citation: Zhang, H.; Yu, L.; Chen, Y.; Wei, Y. Fast Complex-Valued CNN for Radar Jamming Signal Recognition. *Remote Sens.* **2021**, *13*, 2867. https://doi.org/10.3390/rs13152867

Academic Editors: Ming-Der Yang and Alexandre Baussard

Received: 26 May 2021
Accepted: 15 July 2021
Published: 22 July 2021

1. Introduction

Due to the powerful capability to detect, recognize, and track targets under the conditions of all weather, radar is widely used in many weapon systems, and it has become an indispensable electronic equipment in military [1]. With the continuous development of radar jamming technology, many radar jamming technologies have been developed to attack radar systems in order to reduce their target detection, recognition, and tracking capabilities [2]. In addition to various radar jamming methods, more and more electromagnetic signal use in the battlefield environment also seriously affects the effectiveness of the radar system. Radar anti-jamming technology is used for protecting the radar from jamming [3]. As an important prerequisite for effective anti-jamming measures, radar jamming signal recognition has received more and more attention.

Recently, many recognition approaches of radar jamming signals have been proposed, including likelihood-based methods [4–6] and feature-based methods [7–13]. Based on the prior information, the likelihood-based methods identify the type of jamming signal by matching the likelihood function of the jamming signal with some determined thresholds. For instance, based on the adaptive coherent estimator and the generalized likelihood ratio test (GLRT), Greco et al. [4] proposed a deception jamming recognition method. Moreover, in [5], an adaptive detection algorithm based on the GLRT implementation of a generalized Neyman–Pearson rule was proposed to discriminate radar targets and electronic countermeasure (ECM) signals. In addition, Zhao et al. [6] utilized the conventional linear model to discriminate the target and radar jamming signal. Unfortunately, neither the necessary prior information nor the appropriate threshold can be guaranteed in reality, which limits the application of likelihood-based methods for radar jamming signal recognition.

Compared with the signal in the original domain, the extracted features in the transformation domain are usually more separable. Therefore, some jamming signal recognition methods based on feature extraction were proposed. The feature-based method consists of feature extraction and classifier selection. Many researchers extract distinguishing features from jamming signal based on multi-domains, including the time domain [7], frequency domain [8], time–frequency domain [9], and so on. For example, using the product spectrum matrix (SPM), Tian et al. [10] proposed a deception jamming recognition method. Moreover, in [11], support vector machine (SVM) was selected as a classifier to recognize chaff jamming. In [12], based on the AdaBoost.M1 algorithm, a radar jamming recognition algorithm that selected SVM as the component classifiers was proposed. In addition, based on different classifiers (i.e., decision tree, neural network, and support vector machine), Meng et al. [13] extracted five features of jamming signals, including three time-domain features and two frequency-domain features, which could increase the robustness of the recognition system. Nevertheless, the aforementioned methods require a lot of manpower, and high time costs still remain for these approaches based on feature extraction.

Recently, CNN, which is one of the popular deep learning models, has achieved great performance in many fields, including image, text, and speech processing [14]. Due to the uniqueness of the network structure of CNN (e.g., local connections and shared weights), CNN-based methods can automatically extract the invariant and discriminant features of data [15]. Moreover, a growing number of CNN-based methods have been proposed in the field of jamming recognition [16–19]. In [16], a well-designed CNN was utilized to recognize active jamming, and the results indicated that CNNs had the power and ability to distinguish active jamming. In addition, in [17], a CNN-based method was proposed for the recognition of radar jamming, which could recognize nine typical radar jamming signal. By integrating residual block and asymmetric convolution block, Qu et al. [18] proposed a jamming recognition network, which could better extract the features of jamming signals. Moreover, based on the CNN, Wu et al. [19] proposed an automatic radar jamming signal recognition method, which could recognize five kinds of radar jamming signal. Andriyanov et al. used CNN in radar image processing, and they analyzed the construction process of CNN in detail [20]. The recognition rate and recall rate of radar images could reach more than 90%, which illustrated the effectiveness of CNN in processing radar data. Recently, Shao et al. designed a 1D-CNN for radar jamming signal recognition that can identify 12 types of radar jamming with a recognition accuracy of up to 97% [21], which also illustrated the effectiveness of the CNN-based method in processing radar jamming signals.

Radar signals, including radar jamming signals, are naturally complex-valued (CV) based data, including real part and imaginary part. CV data contain more information than RV data, i.e., amplitude r and phase θ. When processing the CV signal, neither the phase nor amplitude information should be ignored.

In order to make full use of the phase and amplitude information of radar jamming signals, a bispectrum analysis was used to keep the phase and amplitude information of the CV signal [22]. Moreover, some researchers have proposed several methods for CV jamming signal recognition [23,24]. For example, based on the amplitude fluctuation, high-order cumulant, and bispectrum, Li et at. studied the feature extraction to analyze and compare the features of deception jamming and target echo [23]. In [24], an algorithm that utilized the bispectrum transformation to extract the amplitude and phase features was proposed to identify deception jamming.

However, most of the existing RV-CNN-based jamming recognition methods mentioned above converted the CV jamming signal into RV and would then input them into RV-CNN for jamming recognition. Since the input and model parameters of the RV-CNN-based methods are RV, they have insufficient significance for phase and amplitude information and are more suitable for processing RV data. For this reason, the loss of phase and amplitude information caused by RV-CNN-based methods is unavoidable, and the problem that RV-CNN cannot fully extract the features of CV data still exists.

Apparently, it is better to extract the CV jamming signal in a complex domain, which can make better use of the unique information of the CV signal, such as phase and amplitude. Therefore, in our work, a radar jamming signal recognition approach based on CV-CNN is proposed. The proposed CV-CNN consists of multiple CV layers, including a CV input layer, several alternations of CV convolutional layers, CV pooling layers, and so on. By extending the entire network parameters to a complex domain, the proposed CV-CNN is able to extract CV features from the jamming signal more efficiently than the aforementioned RV-CNN.

Although a CV-CNN can deal with the CV jamming signal perfectly, due to the complex expansion of the entire model, the CV-CNN still suffers from high computational complexity. Unfortunately, jamming signal recognition tasks have strict requirements for real-time performance, making it difficult to deploy a CV-CNN in the resource-constrained battlefield environments. For deep learning model compression, model pruning [25] is a quite effective method by eliminating redundant parameters. Most model pruning methods [26–28] pruned the less important weight of the model in the filter. Therefore, in our work, motivated by filter pruning, a simple yet effective F-CV-CNN algorithm is proposed for accelerating the CV-CNN.

The main contributions of this study are listed as followed:

1. To address the issue that the existing RV-CNN cannot make good use of CV jamming signal, a CV-CNN is proposed to simultaneously extract the features of real and imagery radar signals, which improves the recognition accuracy of a radar jamming signal.
2. In addition, in view of the high real-time requirements of radar jamming recognition tasks, the fast CV-CNN algorithm is proposed to accelerate the recognition speed of the CV-CNN.

The rest of this paper is organized as follows. The proposed methodology of the CV-CNN for CV jamming signal recognition is provided in Section 2. The proposed fast CV-CNN algorithm process is then shown in Section 3. The details of the experiment, the experimental results and the analysis of results are presented in Section 4, followed by the conclusion presented in Section 5.

2. Proposed CV-CNN for Radar Jamming Signal Recognition

In this section, to address the problem that existing neural network methods cannot make full use of the CV information of radar jamming signal, a novel CV-CNN is proposed. The proposed CV-CNN extends the network parameters from real number to complex number. It is expected that the CV-CNN efficiently extracts the features from the radar jamming signal compared with the existing RV-CNN.

2.1. RV-CNN-Based Radar Jamming Signal Recognition

In this subsection, well-designed single-path RV-CNN (S-RV-CNN) and dual-path RV-CNN (D-RV-CNN) are utilized to extract the abstract features. The structures of the S-RV-CNN and D-RV-CNN are shown in Figures 1 and 2, respectively.

As shown in Figure 1, the S-RV-CNN only contains one channel, and the input is the RV data after the real and imaginary parts of the CV jamming signal are connected. The S-RV-CNN contains an input layer, along with several alternations of the convolutional layers and pooling layers.

In Figure 2, it can be seen that the D-RV-CNN structure consists of two identical S-RV-CNNs—one for extracting the real features and the other for the imaginary features. In the D-RV-CNN, firstly, the real part and imaginary part of the CV jamming signal are fed as inputs into two identical RV-CNNs for each respective part, and the real part and imaginary part features of the CV jamming signal are extracted. Secondly, the extracted real and imaginary part features of the CV radar jamming signal are concatenated, and then are fed as input to the full connection (FC) layer for feature integration. Finally, the

eigenvector is fed as input into the Softmax classifier to obtain the recognition result of the CV radar jamming signal.

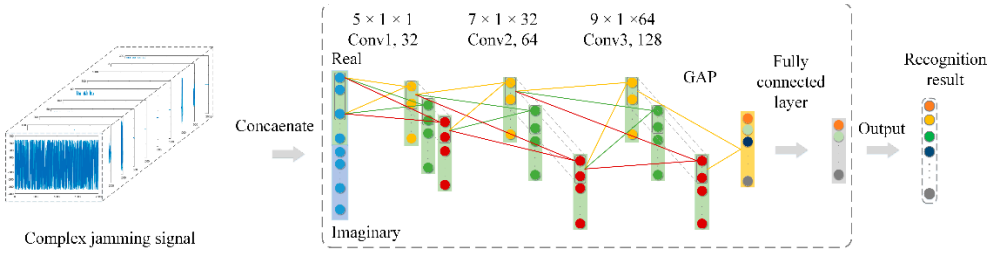

Figure 1. S-RV-CNN structure framework for radar jamming signal recognition.

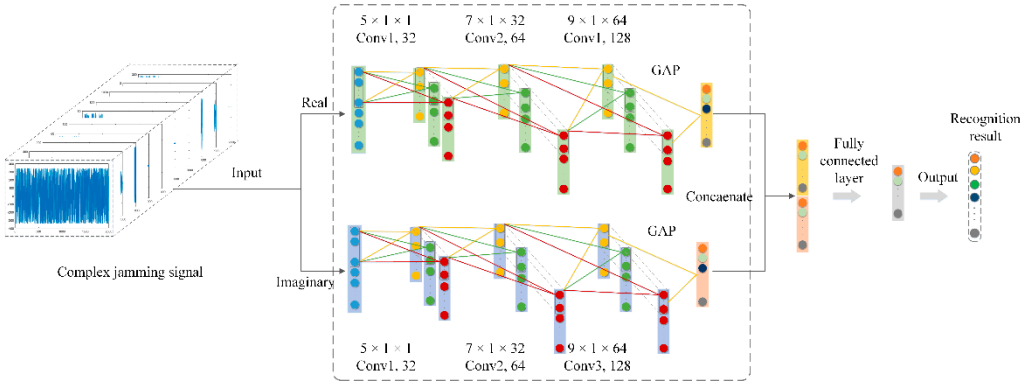

Figure 2. D-RV-CNN structure framework for radar jamming signal recognition. D-RV-CNN contains two identical S-RV-CNNs, i.e., the real part path and the imaginary part path, which are used to extract the real and imaginary features of the CV jamming data.

The convolutional layer is the unique structure of the CNN; it extracts the features of the jamming signal automatically via multiple convolution kernels with different filter sizes. The convolution operation can be defined as:

$$v_j^l = \sum_{i=1}^{M} v_i^{l-1} * w_{ij}^l + b_j^l, \tag{1}$$

where M is the number of input feature maps of jamming signal; matrix v_i^{l-1} denotes the i-th feature map of the $(l-1)$-th layer; the current layer output feature map is v_j^l; $*$ is the convolution operator; and w_{ij}^l and b_j^l are the connection weight and bias, respectively.

In this study, the max-pooling operation was used to reduce the data dimension. In addition, the rectified linear unit (ReLU) activation function was used to extract the nonlinear features of radar jamming signal [12].

Moreover, in order to accelerate the convergence speed of the network, a batch normalization (BN) layer was introduced in the D-RV-CNN [13], which can be defined as follows:

$$\mathrm{BN}(\mathrm{x}) = \gamma \frac{\mathrm{x}^{\mathrm{i}} - \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} \mathrm{x}^{\mathrm{i}}}{\sqrt{\frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} (\mathrm{x}_{\mathrm{i}} - \frac{1}{\mathrm{m}} \sum_{\mathrm{i}=1}^{\mathrm{m}} (\mathrm{x}^{\mathrm{i}}))^2 + \varepsilon}} + \beta, \tag{2}$$

where γ and β are two learnable parameters for feature reconstruction.

In addition, a dropout [14] and global average pooling (GAP) [15] are introduced in the D-RV-CNN to alleviate the phenomenon of overfitting. As a regularization method, dropout inactivates a neuron with a certain probability during CNN training. By averaging the whole feature map, GAP is able to regularize the structure of the whole network to prevent overfitting.

2.2. CV-CNN-based Radar Jamming Signal Recognition

The proposed CV-CNN framework for radar jamming signal recognition is shown in Figure 3. Similar to the aforementioned D-RV-CNN, the CV-CNN consists of multiple cascaded layers, including a CV input layer, several alternations of CV convolution layers and CV pooling layers, etc.

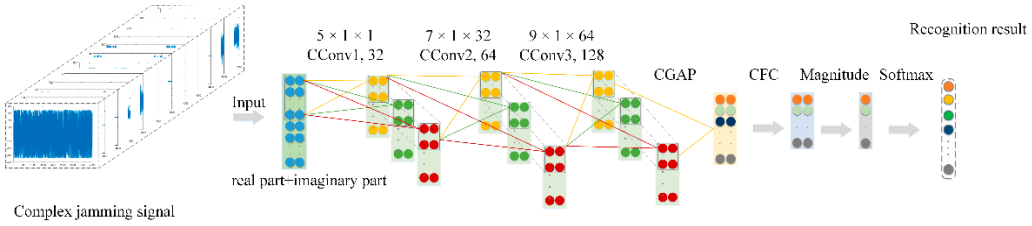

Figure 3. CV-CNN structure framework for radar jamming signal recognition; 5×1×1 CConv1 denotes a 5×1 CV convolution kernel with one channel in the first layer, and 32 is the number of CConv kernels.

The input of the CV-CNN is the CV radar jamming signal, and all mathematical operations of CV-CNN are extended under the theory of complex analysis, including forward propagation and backpropagation. Under the analysis of complex theory, CV-CNN is more suitable for processing a CV jamming signal.

The key structure of the CV-CNN is described below. The first part of the CV-CNN is the CV convolution (CConv) layer. CConv is extended from RV convolution, which is more suitable for extracting deep features of a radar jamming signal. Suppose that $a_k^{(l-1)} \in \mathbb{C}^{W_{l-1} \times H_{l-1} \times I_{l-1}}$ is the input radar jamming signal feature map in the l-th CConv layer, and $\mathbb{C}$ represents the complex domain. In the l-th CConv layer, the number CConv kernel $w_{ik}^{(l)} \in \mathbb{C}^{F_l \times F_l \times I_{l-1} \times I_l}$ is I. The output of CV feature map in the l-th CConv layer is $z_i^{(l)} \in \mathbb{C}^{W_l \times H_l \times I_l}$,

$$\mathrm{z}_\mathrm{i}^{(\mathrm{l})} = \mathrm{f}\left(\mathcal{R}(\mathrm{V}_\mathrm{i}^{(\mathrm{l})})\right) + \mathrm{i} \cdot \left(\mathcal{I}(\mathrm{V}_\mathrm{i}^{(\mathrm{l})})\right), \tag{3}$$

where $f(\cdot)$ represents the non-linear activation function, and $\mathcal{R}(\cdot)$ and $\mathcal{I}(\cdot)$ represent operation of extracting the real part and imaginary part of the CV jamming signal, respectively.

In addition, $V_i^{(l)}$ represents the result of CConv calculation of the CV feature map $a_k^{(l-1)}$ and CConv kernel $w_{ik}^{(l)}$, and the detailed CConv calculation process is shown in Equation (4):

$$\begin{aligned} V_i^{(l)} &= \sum_{\mathrm{k}=1}^{\mathrm{K}} w_{ik}^{(l)} * a_k^{(l-1)} + b_i^{(l)} \\ &= \sum_{k=1}^{K} \mathcal{R}(w_{ik}^{(l)}) * \mathcal{R}(a_k^{(l-1)}) - \mathcal{I}(w_{ik}^{(l)}) * \mathcal{I}(a_k^{(l-1)}) + \mathcal{R}(b_i^{(l)})) \\ &+ i \cdot \sum_{k=1}^{K} \mathcal{R}(w_{ik}^{(l)}) * \mathcal{I}(a_k^{(l-1)}) - \mathcal{I}(w_{ik}^{(l)}) * \mathcal{R}(a_k^{(l-1)}) + \mathcal{I}(b_i^{(l)})). \end{aligned} \tag{4}$$

The second part of the CV-CNN is the CV activation function. In this work, three CV activate functions are extended from RV ReLU (i.e., modReLU [16], zReLU [17], and cReLU).

The first is the modReLU activation function. By introducing a learnable parameter b, modReLU constructs a zero-setting zone with radius b. The expression of modReLU is as follows:

$$\mathrm{modReLU}(\mathrm{z}) = \mathrm{ReLU}(|\mathrm{z}| + b)\mathrm{e}^{\mathrm{i}\theta_z} = \begin{cases} (|\mathrm{z}| + \mathrm{b})\frac{\mathrm{z}}{|\mathrm{z}|} & if \ \ (|\mathrm{z}| + b) \geq 0 \\ 0 & \text{otherwise} \end{cases}, \tag{5}$$

where z is the input radar jamming feature map, and θ_z is the phase of z.

The expression of zReLU is as follows:

$$\mathrm{zReLU(z)} = \begin{cases} z & \text{if } 0 \le \theta_z \le \pi/2 \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

It can be found from Equation (6) that zReLU retains the CV jamming feature map when the phase is between 0 and $\pi/2$.

According to the phase value, the zReLU performs the ReLU operation on the input CV jamming feature map, while the cReLU performs the ReLU operation on the real and imaginary parts of the input CV jamming feature map separately. The expression of cReLU is as follows:

$$\mathrm{cReLU(z)} = \mathrm{ReLU}(\mathcal{R}(\mathrm{z})) + \mathrm{i}\cdot\mathrm{ReLU}(\mathcal{I}(\mathrm{z})) = \max(0, \mathcal{R}(\mathrm{z})) + \mathrm{i}\cdot\max(0, \mathcal{I}(\mathrm{z})). \tag{7}$$

The second part of is CV BN (CBN) layer, which is extended from the real domain to the complex domain. Real-valued BN (RBN) consists essentially of two steps, i.e., standardization, and scaling and shifting. Nevertheless, only transforming and scaling the CV jamming feature map might lead to the deviation of the data distribution. Therefore, it is necessary to perform CBN for the CV jamming signal.

Since a CV jamming signal x consists of real part $\mathcal{R}(x)$ and imaginary part $\mathcal{I}(x)$, we regard $\mathcal{R}(x)$ and $\mathcal{I}(x)$ as two components of x. Then the covariance matrix V can be utilized to normalize x to the normalized $\overline{x}$:

$$\overline{\mathrm{x}} = \mathrm{V}^{-\frac{1}{2}}(\mathrm{x} - \mathrm{E[x]}), \tag{8}$$

and V can be obtained by:

$$\mathrm{V} = \begin{pmatrix} \mathrm{V_{rr}} & \mathrm{V_{ri}} \\ \mathrm{V_{ir}} & \mathrm{V_{ii}} \end{pmatrix} = \begin{pmatrix} \mathrm{Cov}(\mathcal{R}(\mathrm{x}), \mathcal{R}(\mathrm{x})) & \mathrm{Cov}(\mathcal{R}(\mathrm{x}), \mathcal{I}(\mathrm{x})) \\ \mathrm{Cov}(\mathcal{I}(\mathrm{x}), \mathrm{R(x)}) & \mathrm{Cov}(\mathcal{I}(\mathrm{x}), \mathcal{I}(\mathrm{x})) \end{pmatrix}, \tag{9}$$

where $\mathrm{Cov}(\cdot)$ denotes the covariance function. Similar to RBN, two variables (i.e., scaling parameter β and shifting parameter γ) are introduced to restore the learned features. The CBN operation can be defined as:

$$\mathrm{CBN}(\overline{\mathrm{x}}) = \gamma\overline{\mathrm{x}} + \beta, \tag{10}$$

where scaling parameters β is a CV parameter with two learnable parameters, and γ is a 2×2 matrix with three parameters to be learned:

$$\gamma = \begin{pmatrix} \gamma_{\mathrm{rr}} & \gamma_{\mathrm{ri}} \\ \gamma_{\mathrm{ri}} & \gamma_{\mathrm{ii}} \end{pmatrix}. \tag{11}$$

By reducing the correlation between the real part and the imaginary part of the CV jamming signal, CBN can alleviate the overfitting phenomenon.

In addition, the pooling layer, dropout layer, and GAP layer are also extended from the real domain to complex domain. Mathematically, the extension of CV pooling (CPooling), CV dropout (Cdropout), and CV GAP (CGAP) is defined as:

$$\mathrm{CPooling} = \mathrm{Pooling}(\mathcal{R}(\mathrm{z})) + \mathrm{i}\cdot\mathrm{Pooling}(\mathcal{I}(\mathrm{z})), \tag{12}$$

$$\mathrm{CDropout} = \mathrm{Dropout}(\mathcal{R}(z)) + \mathrm{i}\cdot\mathrm{Dropout}(\mathcal{I}(z)), \tag{13}$$

$$\mathrm{CGAP} = \mathrm{GAP}(\mathcal{R}(z)) + \mathrm{i}\cdot\mathrm{GAP}(\mathcal{I}(z)). \tag{14}$$

Then, the output of CGAP becomes the input to the CV FC (CFC) layer. At the end of the CV-CNN, the magnitude of the CV jamming features output by the CFC layer is calculated. The magnitude is taken as the input into the Softmax classifier to obtain the jamming recognition information.

In this paper, we take the weight parameter update process in the CV layer as an example to derive the weight update process. Similar to the RV-CNN, the CV-CNN also utilizes the backpropagation algorithm (BP) algorithm to update the parameters. In this paper, the CE loss function is used to train the CV-CNN. In the BP algorithm, the update formula for weights and deviations is:

$$w_{ik}(t+1) = w_{ik}(t) + \Delta w_{ik}(t) = w_{ik}(t) - \eta \frac{\partial \text{Loss}}{\partial w_{ik}(t)}. \tag{15}$$

The gradient of the loss function can be calculated as:

$$\begin{aligned} \frac{\partial \text{Loss}}{\partial w_{ik}^{(l)}} &= \frac{\partial \text{Loss}}{\partial \mathcal{R}(w_{ik}^{(l)})} + i\frac{\partial \text{Loss}}{\partial \mathcal{I}(w_{ik}^{(l)})} \\ &= \left(\frac{\partial \text{Loss}}{\partial \mathcal{R}(V_i^{(l)})} \frac{\partial \mathcal{R}(V_i^{(l)})}{\partial \mathcal{R}(w_{ik}^{(l+1)})} + \frac{\partial \text{Loss}}{\partial \mathcal{I}(V_i^{(l)})} \frac{\partial \mathcal{I}(V_i^{(l)})}{\partial \mathcal{R}(w_{ik}^{(l)})} \right) \\ &+ i \cdot \left(\frac{\partial \text{Loss}}{\partial \mathcal{R}(V_i^{(l)})} \frac{\partial \mathcal{R}(V_i^{(l)})}{\partial \mathcal{I}(w_{ik}^{(l)})} + \frac{\partial \text{Loss}}{\partial \mathcal{I}(V_i^{(l)})} \frac{\partial \mathcal{I}(V_i^{(l)})}{\partial \mathcal{I}(w_{ik}^{(l)})} \right). \end{aligned} \tag{16}$$

According to Formula (3) and Formula (4), Equation (16) can be simplified as:

$$\frac{\partial \text{Loss}}{\partial w_{ik}^{(l)}} = -\delta_i^{(l)} (a_i^{(l-1)})^*, \tag{17}$$

where $(\cdot)^*$ denotes conjugate calculation, and the $\delta_i^{(l)}$ denotes the error term:

$$\delta_i^{(l)} = -\frac{\partial \text{Loss}}{\partial \mathcal{R}(V_i^{(l)})} - i \cdot \frac{\partial \text{Loss}}{\partial \mathcal{I}(V_i^{(l)})}. \tag{18}$$

Substituting the weight gradient into the Equation (15), we can then complete the update of the weight parameters.

3. F-CV-CNN for Radar Jamming Signal Recognition

The CV-CNN model can make full use of the information of a radar CV jamming signal. Unfortunately, it is difficult to meet the real-time requirements of radar jamming signal recognition due to the high computational complexity. In order to address this issue, a simple yet effective pruning-based fast CV-CNN (F-CV-CNN) method is proposed; the framework of the F-CV-CNN is shown in Figure 4.

As shown in Figure 4, for CV-CNN, some unnecessary CConv kernels that have less impact on the output can be pruned. This can reduce the complexity of the model, so that the CV-CNN can meet the real-time requirement of jamming signal recognition tasks.

Suppose that $\Omega = \{(W^1, b^1), (W^2, b^2), \ldots, (W^L, b^L)\}$ is an over-parameter CV-CNN model for radar jamming signal recognition, where $W \in \mathbb{C}^{F_l \times F_l \times I_{l-1} \times I_l}$ denotes the CConv kernel, and L is the number of CConv layers. On the given CV jamming dataset $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, the difference between the output $\hat{y}$ of CV-CNN and the true label y can be evaluated by the loss function $L(y, \hat{y}, \Omega)$. Therefore, we need to find the subnetwork Ω^* based on the given over-parameter model Ω, and make Ω^* meet the minimum increase in $L(y, \hat{y}, \Omega)$.

In order to obtain Ω^*, we can eliminate the unnecessary parameters of the CV-CNN based on model pruning, which is a simple yet effective method to compress and accelerate the deep learning model.

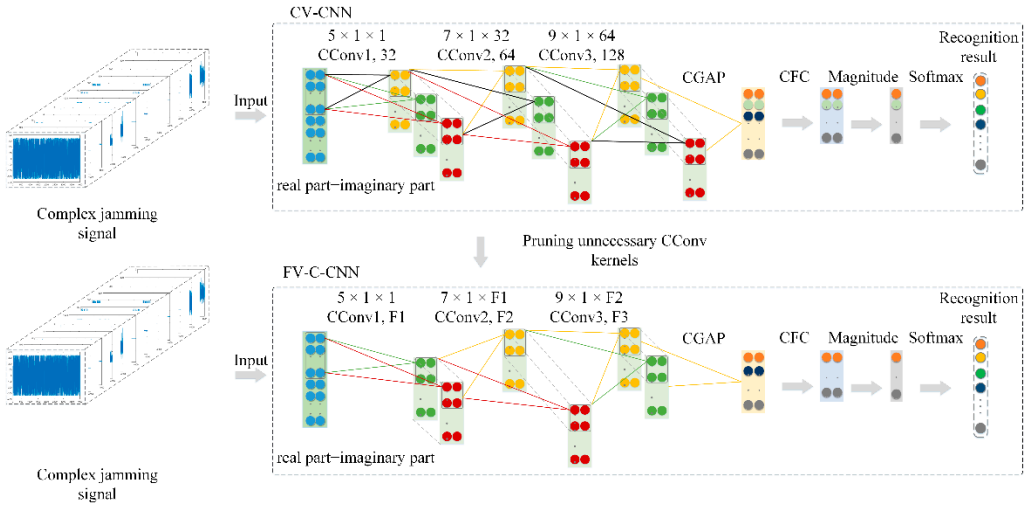

Figure 4. The framework of the F-CV-CNN for radar jamming signal recognition. F1, F2, and F3 each represent the number of remaining CConv kernels after CV-CNN is pruned by the F-CV-CNN algorithm.

For model pruning, a suitable CConv kernel importance evaluation method is vitally important. In this work, considering the parameters of the CV-CNN being in complex form and the real-time requirement of the jamming recognition task, the magnitude of CConv is utilized as the importance evaluation criterion. In actual engineering applications, the computational complexity of the magnitude is low, which is more suitable for engineering implementation. The calculation formula of the magnitude is as follows:

$$\mathrm{P}_{\mathrm{i}}^{(\mathrm{l})} = \sqrt{\left(\mathcal{R}(\mathrm{w}_{\mathrm{i}}^{(\mathrm{l})})\right)^2 + \left(\mathcal{I}(\mathrm{w}_{\mathrm{i}}^{(\mathrm{l})})\right)^2}, \tag{19}$$

where $P_i^{(l)}$ is the magnitude of the i-th CConv kernel of layer l. A smaller $P_i^{(l)}$ means the less importance of the CConv kernel $w_i^{(l)}$, and the impact on the CV-CNN is smaller, so it can be eliminated.

By removing the CConv kernel together with their connection feature maps, the computational cost is significantly reduced. Suppose the input jamming feature map for the l-th CConv layer is $a_k^{(l-1)} \in \mathbb{C}^{W_{l-1} \times H_{l-1} \times I_{l-1}}$. Through the CConv operation $W^{(l)} \in \mathbb{C}^{F_l \times F_l \times I_{l-1} \times I_l}$ in the l-th CConv layer, the input CV jamming feature map $a_k^{(l-1)}$ is then converted into the CV output jamming feature map $a_k^{(l)} \in \mathbb{C}^{W_l \times H_l \times I}$. In the l-th CConv layer, the total number of multiplication operations of CConv is $2 \times F_l \times F_l \times I_{l-1} \times I_l \times W_l \times H_l$. When a certain CConv kernel is eliminated, $2 \times F_l \times F_l \times I_{l-1} \times W_l \times H_l$ multiplication operations are reduced in the l-th CConv layer. In addition, the corresponding output feature maps are also removed, which further reduces $2 \times F_{l+1} \times F_{l+1} \times I_{l+1} \times W_{l+1} \times H_{l+1}$ multiplications in the l+1-th CConv layer.

The F-CV-CNN algorithm is a "train, prune, and fine-tune" algorithm, which means that F-CV-CNN can be summarized into three steps: training, pruning, and fine-tuning.

Figure 1 is divided into three parts: training set, validation set, and test set. The training set and validation set are fed into the over-parameter CV-CNN for model training.

Secondly, in the process of pruning, considering that the layer-by-layer iterative fine-tuning pruning strategy is time-consuming, which is not suitable for real-time jamming signal recognition task, a one-shot pruning and fine-tuning is adopted to prune the over-parameter CV-CNN Ω. The magnitude $P_i^{(l)}$ of the CConv kernel is calculated layer by layer. Next, in the l-th layer, CConv kernels are sorted according to the value of $P_i^{(l)}$. Then, m^l CConv kernels with the smallest value of $P_i^{(l)}$ are pruned, and the corresponding feature

maps are removed simultaneously. The pruning algorithm is iteratively carried out layer by layer until all the CConv layers are pruned, and the simplified model Ω^* is obtained.

Finally, in most cases, due to the model pruning, the simplified model Ω^* will have a performance loss compared to the over-parameter model Ω. Therefore, in order to recover the accuracy loss caused by pruning, we need to fine-tune Ω^*. In our work, we adopted a fine-tune measure to train the simplified model Ω^* for fewer epochs at a lower learning rate. Algorithm 1 shows the workflow of the proposed fast CV-CNN.

Algorithm 1: F-CV-CNN method for radar jamming recognition.

1. **begin**
2. divide the jamming signal set into training set, validation set, and test set,
3. input training set and validation set into CV-CNN to train an over-parameterized model Ω,
4. iteratively pruning over-parameterized model Ω layer by layer:
5. **for** $l = 1, 2, \ldots, L$ do
6. calculate the magnitude $P_i^{(l)}$ of each CConv kernel of l-th layer
7. sort all CCon kernels of the l-th layer, according to $P_i^{(l)}$.
8. remove m^l CConv kernels w_l^m with smaller magnitude in l-th layer,
9. remove the feature maps generated by w_l^m.
10. **end**
11. fine-tune the obtain the pruned model Ω^*.
12. **end**

4. Experiments and Results

4.1. Datasets Description

Because linear frequency modulation (line frequency modulation, LFM) signal has the advantages of low peak power and simple modulation method, it has been widely used in the current pulse compression radar system. The expression of the LFM signal is as follows:

$$x(t) = \text{rect}\left(\frac{t}{T}\right) \exp\left(j\pi\frac{B}{T}t^2\right). \tag{20}$$

Then, 12 kinds of expert-simulated radar jamming signal as experimental datasets were considered to verify the effectiveness of the proposed method, including pure noise, active jamming, and passive jamming. Among them, active jamming includes typical suppression jamming, such as aiming jamming, blocking jamming, sweeping jamming; typical deception jamming such as interrupted sampling repeater jamming (ISRJ) [29]; distance deception jamming [30]; dense false target jamming [31]; smart noise jamming; and passive jamming, such as chaff jamming [11]. In addition, we also considered some composite jamming, such as ISRJ and chaff compound jamming, dense false target and smart noise compound jamming, and distance deception and sweeping compound jamming.

For the simulated radar jamming signal set, each type of radar jamming signal had 500 simulated samples. Each sample had 2000 sampling points, and each sampling point was composed of a real part and imaginary part. Figure 5 shows the waveform of the radar jamming signal in the time domain. The pulse width T of the simulated LFM signal we used was 20 μs , the bandwidth B was 10 MHz, and the sampling frequency f_s was 20 MHz. The simulation parameters of the jamming signal were shown in Table 1.

At the same time, we constructed the expert statistical feature (SF) dataset of the radar jamming signal for the comparative experiment, including four time domain features (i.e., skewness, kurtosis, instantaneous amplitude reciprocal, and envelope fluctuation) and four frequency domain features (i.e., additive gaussian white noise factor, maximum value of normalized instantaneous amplitude and frequency, instantaneous phase correlation coefficient, and frequency stability).

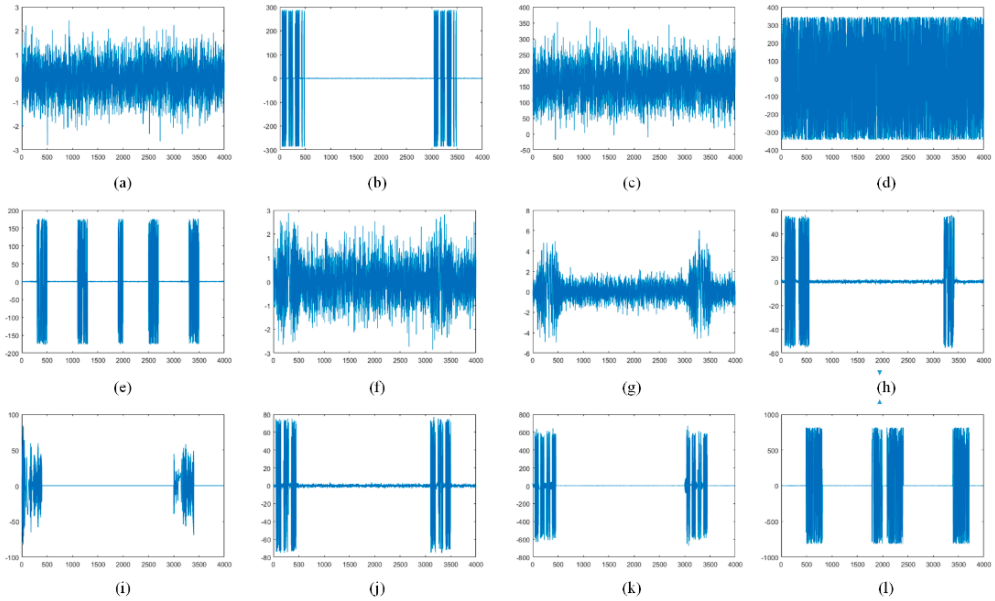

Figure 5. CV radar jamming signal (real points + imaginary points) illustration. On the X-axis, the first 2000 points represent the real part of the CV jamming signal, and the last two thousand points represent the imaginary part of the CV jamming signal. The Y-axis refers to the amplitude values of the real and imaginary parts of the jamming signal. From (**a**) to (**l**): (**a**) pure noise, (**b**) ISRJ, (**c**) aiming, (**d**) blocking, (**e**) sweeping, (**f**) distance deception, (**g**) dense false target, (**h**) smart noise, (**i**) chaff, (**j**) chaff and ISRJ compound, (**k**) dense false target and smart noise compound, (**l**) distance deception and sweeping compound.

Table 1. Parameter setting of radar jamming signal simulation.

Jamming	Parameters	Value Range
ISRJ	JNR (dB) Sampling duration (μs) Pulse repetition times	30 to 60 20 to 40 30 to 60
Aiming	JNR (dB) Bandwidth (MHz)	50 to 80 30 to 60
Blocking	JNR (dB) Bandwidth (MHz)	10 4×10^{-5} to 8×10^{-5}
Swept	JNR (dB) Sweep range (MHz) Sweep cycle	1 to 10 0.5 to 2 30 to 60
Distance deception	False target delay range (μs) False target range	1 to 4 1 to 3
Smart noise	JNR (dB) Sampling duration (μs) Pulse repetition times	3 to 5 1 to 10 0.5 to 2
Dense false target	Number of false targets False target delay range (μs) False target range	1000 to 2000 20 to 50 10 to 20
Chaff	Number of chaff Average Doppler frequency Doppler parity variance	30 to 60 20 to 40 30 to 60

4.2. Experimental Setup

In this study, in order to fully verify the effectiveness of the proposed CV-CNN, the well-designed S-RV-CNN and D-RV-CNN were used for the comparative experiments. The

D-RV-CNN consisted of two identical S-RV-CNNs that could extract the real and imaginary features of the CV jamming signal separately. The detailed network structure of RV-CNN is shown in Table 2.

Table 2. The architecture of the RV-CNN.

Method	Convolution	BN	ReLU	Pooling	Dropout
S-RV-CNN			Input layer		
	CConv (5,1,32)	CBN	cReLU	CMaxPooling (2×2)	No
	CConv (7,32,64)	No	cReLU	CMaxPooling (2×2)	No
	CConv (9,64,128)	No	cReLU	CMaxPooling (2×2)	CDropout (50%)
			GAP		
			FC		
			Output layer		
D-RV-CNN			Input layer		
	CConv (5,1,32)	CBN	cReLU	CMaxPooling (2×2)	No
	CConv (7,32,64)	No	cReLU	CMaxPooling (2×2)	No
	CConv (9,64,128)	No	cReLU	CMaxPooling (2×2)	CDropout (50%)
			GAP		
			Concatenate real and imaginary part features		
			FC		
			Output layer		

The S-RV-CNN contained three convolutional layers, three pooling layers, a dropout layer, a BN layer, and a GAP layer. The activation function adopted the ReLU activation function. The convolution step lengths were all set to 1, and the pooling operation adopted a max-pooling operation.

The D-RV-CNN consisted of two identical S-RV-CNNs. The inputs of the two S-RV-CNNs were the real and imaginary parts of the CV jamming signal, which were used to extract the features of the real and imaginary parts separately. The concatenation layer was used to concatenate the real and imaginary part features extracted by two S-RV-CNNs, and then sent the concatenated feature vector as input into the FC layer for feature integration. Finally, the recognition result was obtained by Softmax.

For the proposed CV-CNN, its specific network structure is shown in Table 3. Extending the RV-CNN to the complex domain, the CV-CNN included three CConv layers and CPooling layers, one CDropout layers, a CBN layer, and a CGAP layer. The activation function was also extended from the real domain to the complex domain, adopting the ReLU activation function in complex form.

Table 3. The architecture of the proposed CV-CNN.

Convolution	BN	ReLU	Pooling	Dropout
		Input layer		
CConv (5,1,32)	CBN	cReLU	CMaxPooling (2×2)	No
CConv (7,32,64)	No	cReLU	CMaxPooling (2×2)	No
CConv (9,64,128)	No	cReLU	CMaxPooling (2×2)	CDropout (50%)
		CGAP		
		CFC		
		Magnitude layer		
		Output layer (Softmax)		

We divided the CV radar jamming signal used in the experiment into three subsets: training set, validation set, and test set. For each class of radar jamming, we randomly selected 25, 50, 75, 100, and 150 samples per class as the training set, 50 samples per class as the validation set, and the remaining data as the test set.

For the S-RV-CNN, D-RV-CNN, and the CV-CNN, we kept the same hyperparameter settings during the experiment. The learning rate was annealed down from 0.009 to 0.0008, and the batch size and training epochs were set to 64 and 100, respectively.

In addition, we used the SF dataset to train different classifiers as the comparison experiment, i.e., support vector machine (SVM), random forest (RF) [32], decision tree (DT) [33], and K-nearest neighbors (KNN) [34]. The kernel function of the SVM was a radial basis kernel function, the maximum tree depth of the DT was set to 6, the size of the RF was set to 300, and K in KNN was set to 10.

To evaluate the effectiveness of the proposed methods, we used two evaluation metrics, i.e., overall accuracy (OA) and kappa coefficient (K), to evaluate the recognition effect of different methods. OA could be calculated as:

$$\mathrm{OA} = \frac{\mathrm{n}}{\mathrm{N}} \times 100, \tag{21}$$

The kappa coefficient could be calculated as:

$$\begin{cases} \mathrm{p_e} = \frac{\mathrm{a_1 \times b_1 + a_2 \times b_2 + \cdots + a_{12} \times b_{12}}}{\mathrm{N^2}} \\ \mathrm{Kappa} = \frac{\mathrm{OA - p_e}}{1 - \mathrm{p_e}} \end{cases} \tag{22}$$

where $a_1, a_2, \ldots, a_{12}$ denoted the number of test samples for each class jamming signal, and $b_1, b_2, \ldots, b_{12}$ denoted the number of correct recognitions of each class jamming signal.

In our work, CV-CNN training and testing, pruning, and other comparative experiments were run on a computer equipped with a 3.4-GHz CPU and a NVIDIA GeForce GTX 1070ti card. All deep learning methods were implemented on the PyCharm Community 2019.1 platform with Pytorch 1.1.0 and CUDA 9.0. In addition, F-CV-CNN was also verified on the NVIDIA Jetson Nano development kit.

4.3. Experimental Results and Performance Analysis

4.3.1. The Recognition Results of CV-CNN

For the proposed CV-CNN, we first considered the impact of different activation function layers and different BN layers on its recognition effect. In this work, three CV activation functions were utilized to explore the influence of different activation functions on the recognition effect of the CV-CNN.

Figure 6 shows the recognition effect of the CV-CNN with different activation functions, i.e., modReLU, zReLU, and cReLU. As shown in Figure 6, under the condition of 50 training samples of each class, the CV-CNN could obtain the recognition accuracy of 93.45%, 96.78%, and 97.14% when using the modReLU, zReLU, and cReLU activation functions, respectively. When the activation function was cReLU, the proposed CV-CNN could achieve the best recognition performance.

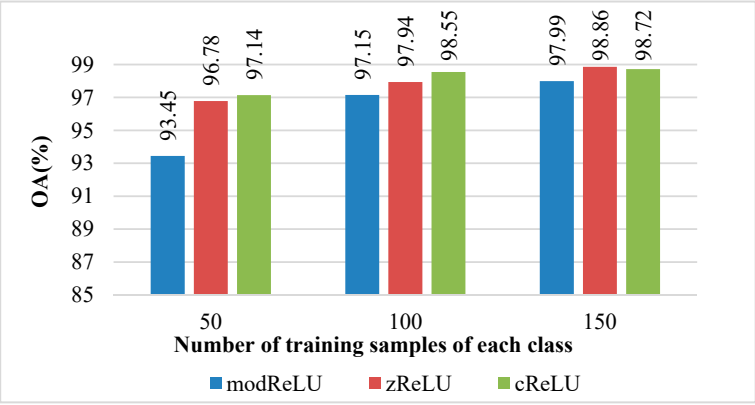

Figure 6. Recognition results of CV-CNN with different activate functions.

The recognition effect of the CV-CNN when adopting different BN layers is shown in Table 4. As shown in Table 4, when the BN layer was not introduced in CV-CNN, and when RBN and CBN were used with CNN separately, recognition accuracies of 96.75%, 96.92% and 97.14% can be obtained, respectively. When using the CBN layer, CV-CNN could achieve the best recognition effect.

Table 4. Recognition results of CV-CNN with different BN layer.

N	Method	CV-CNN	CV-CNN-RBN	CV-CNN-CBN
50	OA (%)	96.75±0.64	96.92±0.55	**97.14±0.70**
	K × 100	96.46±0.70	96.64±0.60	**96.88±0.76**
100	OA (%)	97.39±0.77	98.10±0.40	**98.55±0.34**
	K × 100	97.48±0.84	97.92±0.43	**98.42±0.37**
150	OA (%)	98.33±0.30	98.43±0.48	**98.72±0.19**
	K × 100	98.18±0.33	98.29±0.51	**98.61±0.21**

Similar experimental results could be obtained when each class of training sample was 100 and 150. For CV-CNN, using the cReLU activation function and a CBN layer could achieve better recognition accuracy.

Table 5 shows the detailed results of the proposed CV-CNN, and the best recognition accuracy is highlighted in bold. In Table 4, C1–C12 represent pure noise, ISRJ, aiming jamming, blocking jamming, sweeping jamming, distance deception jamming, dense false target jamming, smart noise, chaff jamming, chaff and ISRJ compound jamming, dense false target and smart noise compound jamming, distance deception, and sweeping compound jamming, respectively. From Table 4, one can see that CV-CNN demonstrated its best recognition performance in terms of OA and K.

Table 5. Recognition results on radar jamming signal dataset (values ± standard deviation). The best accuracy is highlighted in bold.

N	Method	SF-RBF-SVM	SF-RF-300	SF-DT	SF-KNN	S-RV-CNN	D-RV-CNN	CV-CNN
25	OA(%)	73.44 ± 1.29	90.26 ± 0.42	87.80 ± 1.08	69.40 ± 0.36	84.85 ± 1.74	84.05 ± 1.25	**95.49 ± 1.35**
	K×100	71.02 ± 1.41	89.38 ± 0.46	86.69 ± 1.18	66.62 ± 0.39	83.48 ± 1.90	82.61 ± 1.37	**95.08 ± 1.47**
50	OA(%)	79.17 ± 1.33	90.52 ± 0.10	88.67 ± 0.70	71.51 ± 0.86	90.17 ± 1.93	91.43 ± 0.88	**97.14 ± 0.70**
	K×100	77.28 ± 1.45	89.65 ± 0.11	87.64 ± 0.77	68.93 ± 0.94	89.27 ± 2.11	90.65 ± 0.97	**96.88 ± 0.76**
75	OA(%)	80.93 ± 0.66	90.71 ± 0.40	89.84 ± 0.29	72.47 ± 0.32	90.79 ± 1.54	93.24 ± 1.71	**98.24 ± 0.33**
	K×100	79.20 ± 0.72	89.87 ± 0.44	88.92 ± 0.31	69.97 ± 0.34	89.96 ± 1.68	92.63 ± 1.86	**98.08 ± 0.36**
100	OA(%)	82.46 ± 0.29	91.52 ± 0.12	89.27 ± 0.17	72.88 ± 0.53	94.20 ± 1.39	95.24 ± 0.86	**98.55 ± 0.34**
	K×100	80.86 ± 0.32	90.75 ± 0.13	88.30 ± 1.8	70.42 ± 0.58	93.67 ± 1.52	94.81 ± 0.94	**98.42 ± 0.37**
150	OA(%)	83.56 ± 0.26	91.63 ± 0.18	89.89 ± 0.71	73.30 ± 0.52	96.10 ± 1.28	96.27 ± 0.22	**98.72 ± 0.19**
	K×100	82.07 ± 0.29	90.87 ± 0.19	88.97 ± 0.77	70.87 ± 0.57	95.75 ± 1.40	95.93 ± 0.25	**98.61 ± 0.21**
	C1	99.66 ± 0.67	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	**100 ± 0.00**	99.93 ± 0.13	**100.00 ± 0.00**
	C2	76.20 ± 2.5	96.80 ± 1.34	90.97 ± 2.64	26.45 ± 3.38	97.93± 3.81	98.27 ± 1.51	**98.93 ± 0.53**
	C3	99.53 ± 0.16	**100.00 ± 0.00**	**100.00 ± 0.00**	99.26 ± 0.14	**100 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**
	C4	98.47 ± 1.36	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**	**100 ± 0.00**	99.87 ± 0.16	99.73 ± 0.39
	C5	98.07 ± 1.31	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**	90.87 ± 3.04	92.47 ± 3.02	98.27 ± 0.83
	C6	97.73 ± 1.06	97.54 ± 1.41	**98.51 ± 0.49**	98.46 ± 0.97	94.07 ± 2.66	94.20 ± 3.15	95.13 ± 2.09
	C7	**98.20 ± 0.62**	97.09 ± 0.33	95.94 ± 0.94	84.45 ± 1.67	94.07 ± 3.97	92.13 ± 2.54	96.93 ± 1.54
	C8	46.27 ± 5.99	62.97 ± 2.49	54.97 ± 4.87	21.26 ± 2.33	93.67 ± 4.42	97.06 ± 1.51	**99.73 ± 0.25**
	C9	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**	94.69 ± 1.15	**100 ± 0.00**	**100.00 ± 0.00**	98.40 ± 1.02
	C10	39.93 ± 7.21	56.46 ± 2.92	54.69 ± 1.99	23.09 ± 2.42	95.27 ± 4.31	94.47 ± 2.65	**99.73 ± 0.24**
	C11	51.60 ± 5.24	88.74 ± 2.84	83.60 ± 2.37	33.26 ± 3.34	**98.73 ± 1.50**	98.73 ± 0.88	98.60 ± 0.77
	C12	97.07 ± 1.97	**100.00 ± 0.00**	**100.00 ± 0.00**	98.69 ± 1.00	88.6 0 ± 4.83	88.07 ± 3.15	99.47 ± 0.62

Under the condition of 25 training samples of each class, the CV-CNN improved the OA by 10.11% compared to the D-RV-CNN. In addition, the CV-CNN also outperformed the SF-classifier methods. In the case of 50 training samples per class, the CV-CNN exhibited the OA over D-RV-CNN and SF-RF, with improvements of 5.73% and 6.64%.

In addition, experiments with 75, 100, and 150 training samples of each class were also conducted to further demonstrate the effectiveness of our method, the results are also shown in Table 5. It could be seen from Table 5 that the CV-CNN achieved the best recognition performance, with 75, 100, and 150 training samples per class, which fully demonstrated the effectiveness of our proposed CV-CNN.

Table 6 shows the results of the proposed method and some previous CNN-based comparative methods. For the detailed network structure of 2D-CNN and 1D-CNN, please refer to [21].

Table 6. Comparison of recognition accuracy between CV-CNN and other CNN-based methods. The best accuracy is highlighted in bold.

N	Method	S-RV-CNN	D-RV-CNN	2D-CNN [21]	1D-CNN [21]	CV-CNN
50	OA (%)	90.17 ± 1.93	91.43 ± 0.88	87.83 ± 1.84	91.95 ± 2.19	**97.16 ± 0.53**
	K × 100	89.27 ± 2.11	90.65 ± 0.97	86.72 ± 2.01	91.21 ± 2.39	**96.90 ± 0.58**
100	OA (%)	94.20 ± 1.39	95.24 ± 0.86	89.88 ± 0.47	96.97 ± 0.88	**98.24 ± 0.36**
	K × 100	93.67 ± 1.52	94.81 ± 0.94	88.97 ± 0.52	96.69 ± 0.96	**98.08 ± 0.39**
150	OA (%)	96.10 ± 1.28	96.27 ± 0.22	91.33 ± 1.29	97.34 ± 0.37	**98.76 ± 0.23**
	K × 100	95.75 ± 1.40	95.93 ± 0.25	90.55 ± 1.40	97.10 ±0.41	**98.64 ± 0.25**

With 50 training samples per class, the CV-CNN achieved a recognition accuracy of 97.16%, which improved by 9.33% and 5.21% in terms of OA compared with the 2D-CNN and 1D-CNN, respectively. With 100 training samples per class, the CV-CNN could achieve a recognition accuracy of 98.24%. Compared with the 2D-CNN (89.88%) and 1D-CNN (96.97%) in [21], the CV-CNN achieved an improvement of 8.36% and 1.27%, respectively. In the case of 150 training samples per class, CV-CNN also achieved the highest recognition accuracy. Compared with 2D-CNN and 1D-CNN, the proposed CV-CNN improved the recognition accuracy of 7.43% and 1.43%, respectively.

Under different Dropout rates, the recognition accuracy of CV-CNN was shown in the Table 7. It could be found that when the dropout rate was small, there was an over-fitting phenomenon for CV-CNN, which led to a decrease in the recognition accuracy of CV-CNN. When the dropout rate was high, the recognition accuracy of CV-CNN would also decrease due to the high proportion of randomly discarded neurons. Therefore, a moderate Dropout rate should be selected. In our experiment, we set the dropout rate to 0.5.

Table 7. CV-CNN recognition accuracy under different dropout rates. The best accuracy is highlighted in bold.

N	Rate	0.1	0.3	0.5	0.7	0.9
50	OA (%)	96.33 ± 0.87	97.10 ± 0.86	97.14 ± 0.70	**97.41 ± 0.56**	96.99 ± 0.61
	K × 100	96.00 ± 0.94	96.84 ± 0.94	96.88 ± 0.76	**97.18 ± 0.61**	96.72 ± 0.66
100	OA (%)	98.09 ± 0.85	98.41 ± 0.44	**98.55 ± 0.34**	98.36 ± 0.25	97.86 ± 0.22
	K × 100	97.91 ± 0.09	98.27 ±0.48	**98.42 ± 0.37**	98.21 ± 0.27	97.67 ± 0.24

It could be found from Figure 7 that CV-CNN can achieve a better recognition of jamming signal, but it is easy to confuse C6 (distance false targets) and C7 (dense false targets). The reason for the above phenomenon was mainly due to the fact that the simulation parameters of distance false targets and dense false targets were the same except for the number of false targets. As a result, the two types of jamming signal had a certain degree of similarity, and it is easy to confuse the two.

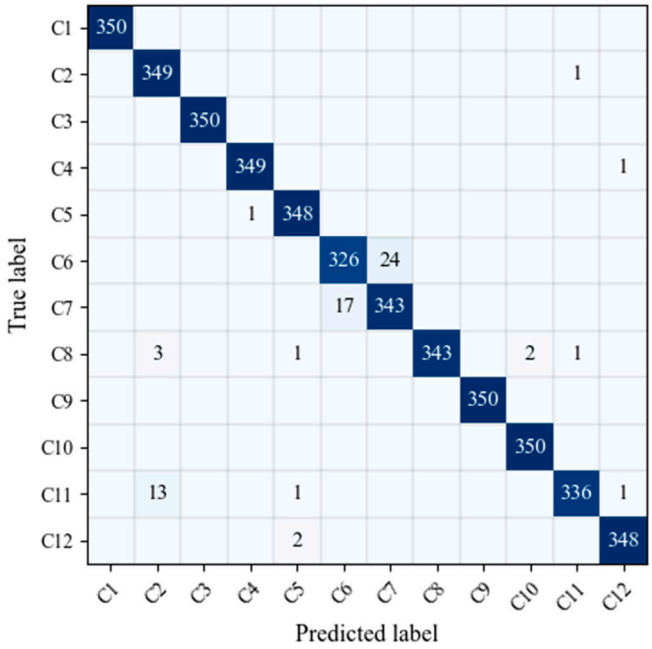

Figure 7. Confusion matrix for the proposed CV-CNN with 150 training samples of each class.

4.3.2. The Recognition Results of F-CV-CNN

Considering that the CConv kernels in different layers of CV-CNN had different sensitivity to pruning, we first analyzed the pruning sensitivity of CV-CNN. The pruning sensitivity curves of different CConv layers in CV-CNN are shown in Figure 8, with 50 and 100 training samples per class. As shown in Figure 8, for the first layer of CConv, when the pruning rate was 14.19%, the model performance after pruning began to decline; for the third layer, when the pruning rate reached 62.77%, the model performance after pruning began to decline. In summary, for the CV-CNN used in the experiment, the pruning sensitivity of the second layer was smaller than that of the first and third layers. In the same CConv layer, the CV-CNN that was trained by more training samples had stronger pruning robustness.

In this subsection, the over-parameter CV-CNN and the simplified model obtained by the F-CV-CNN algorithm were expressed as Ω and Ω^*, respectively. First, with 50, 100, and 150 training samples for each class, Ω was trained separately. Then the F-CV-CNN algorithm was carried out on the Ω to obtain Ω^*.

Table 8 shows the recognition results of Ω^* under different pruning rates. As shown in Table 8, when the pruning ratio is low, the performance loss caused by model pruning can be ignored. However, with the increase of the pruning rate, the recognition performance of Ω^* continues to decline. When there are 50 training sample for each class and the pruning ratio is 85%, the recognition accuracy of Ω^* was reduced from 97.14% to 73.75% compared with Ω. Similarly, when the number of training samples for each class was 100 and 150, the recognition accuracy of Ω^* also dropped to 81.25% and 94.42%, respectively. From Table 8, one could also find the same conclusions as the previous analysis, i.e., the more training samples, the smaller performance loss of Ω^*.

In order to restore the performance of Ω^*, a fine-tune post-processing operation was required. The specific operation was to train 15 epochs at a low learning rate of 0.0001. The experimental results of fine-tune post-processing on Ω^* are also shown in Table 8. As shown in Table 8, the fine-tune post-processing could restore the recognition performance of Ω^* to a certain extent.

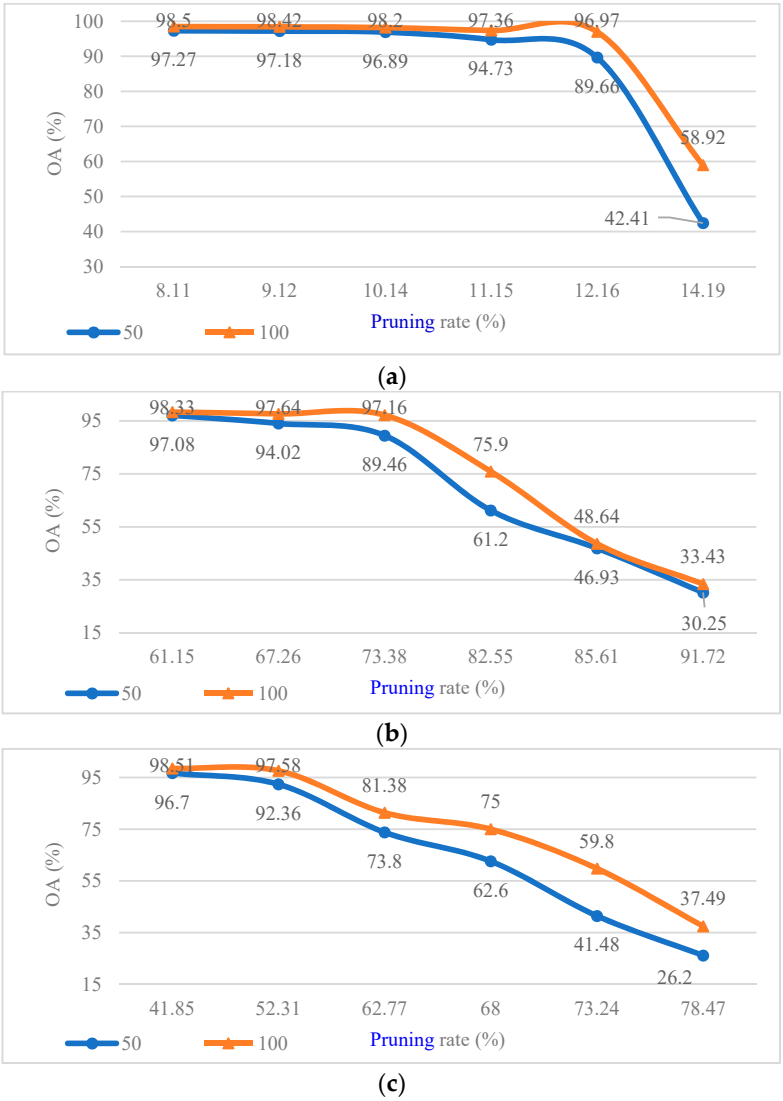

Figure 8. Pruning sensitivity curve of CConv layer of CV-CNN: (**a**) first layer, (**b**) second layer, (**c**) third layer.

Table 8. Recognition results of F-CV-CNN on radar jamming signal dataset (values ± standard deviation).

N	Pruning Rate / Fine-tune	0%	42%	56%	70%	74%	85%	92%
50	No	97.14 ± 0.70	96.74 ± 1.37	96.74 ±1.28	96.73 ± 1.33	96.89 ± 1.34	73.75 ± 15.14	68.3 ± 14.17
	Yes	97.14 ± 0.70	97.47 ± 0.64	97.50 ± 0.66	97.42 ± 0.84	97.59 ± 0.72	93.85 ± 4.27	90.66 ± 5.65
100	No	98.55 ± 0.34	98.54 ± 0.30	98.48 ± 0.34	98.53 ± 0.41	98.36 ± 0.46	81.25 ± 8.66	83.03 ± 9.29
	Yes	98.55 ± 0.34	98.75 ± 0.20	98.78 ± 0.10	98.76 ± 0.14	98.72 ± 0.21	98.27 ± 0.81	97.85 ± 0.99
150	No	98.72 ± 0.19	98.62 ± 0.30	98.73 ± 0.15	98.78 ± 0.17	98.72 ± 0.13	94.42 ± 4.23	94.22 ± 4.20
	Yes	98.72 ± 0.19	98.91 ± 0.19	98.90 ± 0.22	98.94 ± 0.21	98.94 ± 0.25	98.78 ± 0.30	98.61 ± 0.19

With a pruning rate of 92% and 50 training samples per class, the recognition performance of Ω^* after fine-tune processing was restored to 90.66%. Compared with Ω, the performance of Ω^* had dropped by 6.48%. However, in the case of 100 and 150 training samples per class, the performance of Ω^* at a pruning rate of 92% could also be restored well through fine-tune processing. The results also showed that for the deep learning model, the more training samples, the stronger the robustness of the model.

In order to better show the effectiveness of the F-CV-CNN algorithm, we tested the inference time of Ω^* on NVIDIA Jetson Nano, which is used for real-time jamming recognition in our work. Table 9 showed the floating-point operations (FLOPs) and inference time.

Table 9. The FLOPs and inference time for F-CV-CNN.

Pruning Rate	0%	42%	56%	70%	74%	85%	92%
FLOPs (M)	202.85	130.75	98.48	66.21	51.22	33.17	17.00
Nano time (ms)	87.76	79.59	74.25	57.53	41.17	33.38	28.91

With the increase of the pruning rate, the FLOPs of Ω^* continued to decrease, and the inference time also continued to decrease. When the pruning rate was 92%, the FLOPs of Ω^* were reduced by 91% compared with the Ω, and the inference time was also reduced by 67%. The experimental results also fully illustrated the effectiveness of the F-CV-CNN algorithm.

4.3.3. The Recognition of Radar Jamming and Normal Signal

In real application, it is necessary to recognize the radar normal signal and jamming signal at the same time. Therefore, in this subsection, the effectiveness of the proposed CV-CNN was tested on a dataset which contained radar jamming and normal signals.

The radar jamming signals were the signals in Section 4.1, which contained twelve types of radar jamming signals. Furthermore, the normal radar signals were generated based on Equation (20), and they were used as the 13th type of the radar jamming and normal signal dataset. The pulse width T was 20 μs, and bandwidth B was 10 MHz, and the sampling rate was 20 MHz. For the normal radar signals, there were 500 simulated samples. Each sample had 2000 sampling points, and each sampling point was composed of a real part and an imaginary part.

Table 10 shows the experimental results of the proposed method to perform recognition for normal radar signals and jamming signals, where C13 represents the normal radar signal. From Table 10, one can see that CNN-based methods (i.e., S-RV-CNN, D-RV-CNN, and CV-CNN) obtained better recognition performance compared with tradition machine learning methods (i.e., random forest and decision tree). For CNN-based methods, with 50 training samples for each class, the recognition accuracy of the CV-CNN for normal radar signals was 89.60%, which was 5.6% and 3.75% higher than that of the S-RV-CNN (84.00%) and D-RV-CNN (85.85%), respectively.

Table 10. Recognition results on radar jamming and normal signal dataset (values ± standard deviation). The best accuracy is highlighted in bold.

N	Method	RF-300	DT	S-RV-CNN	D-RV-CNN	CV-CNN
50	OA (%)	75.99 ± 0.99	58.83 ± 1.44	88.33 ± 2.06	91.30 ± 2.35	**95.92 ± 0.65**
	K×100	73.99 ± 1.05	55.40 ± 1.56	87.35 ± 2.23	90.57 ±2.54	**95.58 ± 0.70**
	C13	58.04 ± 5.97	35.20 ± 3.36	84.00 ± 4.56	85.85 ±5.00	**89.60 ± 4.07**
100	OA (%)	80.48 ± 0.16	64.17 ± 0.53	93.86 ± 1.00	94.14 ± 1.16	**97.66 ±0.62**
	K×100	78.86 ± 0.18	61.18 ± 0.57	93.35 ± 1.08	93.65 ± 1.26	**97.47 ± 0.67**
	C13	65.60 ± 1.91	42.70 ± 2.56	87.43 ± 3.25	89.94 ± 2.11	**91.20± 3.40**
150	OA (%)	82.28 ± 0.39	67.09 ± 1.06	94.83 ± 1.48	95.44 ± 1.25	**98.18 ± 0.53**
	K×100	80.80 ± 0.42	64.35 ± 1.15	94.40 ± 1.60	95.07 ± 1.35	**98.03 ± 0.58**
	C1	**100.00 ± 0.00**	69.20 ± 5.10	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**
	C2	64.91 ± 0.78	40.29 ± 5.52	**99.94 ± 0.44**	99.73 ± 0.39	99.07 ± 0.61
	C3	**100.00 ± 0.00**	99.20 ± 0.49	**100.00 ± 0.00**	**100.00 ± 0.00**	**100.00 ± 0.00**
	C4	**100.00 ± 0.00**	84.51 ± 1.57	99.87 ±0.27	**100.00 ± 0.00**	**100.00 ± 0.00**
	C5	**100.00 ± 0.00**	97.37 ± 0.71	90.53 ± 4.48	87.93 ± 8.54	99.00 ± 0.37
	C6	81.94 ± 1.59	44.57 ± 4.92	91.67 ± 4.26	90.80 ± 0.95	**92.67 ± 3.11**
	C7	77.14 ± 2.13	39.71 ± 3.31	95.07 ± 3.90	90.47 ± 2.94	**95.93 ± 1.48**
	C8	43.03 ± 1.55	35.43 ± 2.18	88.33 ± 6.20	93.73 ± 5.57	**99.33 ± 0.42**
	C9	**100.00 ± 0.00**	96.06 ± 0.55	99.73 ± 0.39	99.93 ± 0.13	99.93 ± 0.13
	C10	36.86 ± 4.54	33.49 ± 2.55	89.33 ± 6.67	94.33 ± 2.23	**99.93 ± 0.47**
	C11	**99.71 ± 0.26**	87.94 ± 4.20	99.07 ± 0.61	97.73 ± 1.85	97.20 ± 2.19
	C12	**100.00 ± 0.00**	96.57 ± 1.39	87.67 ± 3.09	92.93 ±2.31	99.13 ± 0.16
	C13	66.00 ± 1.43	47.89 ± 2.80	92.13 ± 2.09	93.20 ± 3.64	**94.80 ± 1.55**

Moreover, with 100 training samples per class, the CV-CNN improved the recognition accuracy by 3.77% and 1.26% compared with the S-RV-CNN and D-RV-CNN, respectively. With 150 training samples per class, the CV-CNN also achieved the highest recognition accuracy, and the recognition accuracy for normal radar signals reached 94.80%, which further verified the effectiveness of the proposed CV-CNN.

5. Conclusions

In this paper, we proposed a CV-CNN for radar jamming signal recognition which extends the network parameters from real number to complex number. Through CConv, CPooling, CBN layer, and so on, the unique information of the CV jamming data can be better utilized. Experimental results show that our proposed method can better identify CV jamming signals, and its recognition performance was further improved, with the improvements of 11.44% in terms of OA when there are 25 training samples per class.

In addition, in order to deal with the issue of over parameters and poor real-time performance, the F-CV-CNN algorithm was proposed. Based on filter-level filter pruning, unnecessary CConv kernels were pruned to reduce the number of parameters and calculations of CV-CNN so that the CV-CNN meets the real-time requirements of interference signal recognition tasks. After pruning the F-CV-CNN, FLOPs can be reduced by up to 91% compared with CV-CNN, and the inference time can be reduced by 67%.

Furthermore, the effectiveness of the proposed CV-CNN was also tested on a dataset which contained radar jamming and normal signals. The results showed that the CV-CNN worked well on the recognition of radar jamming and normal signals at the same time.

For radar jamming signal recognition, we usually face a problem of limited training samples. Due to a large number of learnable parameters, this problem is serious when deep learning-based methods are used for radar jamming signal recognition. Therefore, in the near future, deep CV CNN-based recognition with limited training samples is the next move of our research.

Author Contributions: Conceptualization, Y.C.; methodology, H.Z., L.Y., Y.C. and Y.W; writing—original draft preparation, H.Z., L.Y., Y.C. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of China under the Grant 61971164 and U20B2041.

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

References

1. Liu, N.; Zhang, Y. A survey of radar ECM and ECCM. *IEEE Trans. Aerosp. Electron. Syst.* **1995**, *27*, 1110–1120.
2. Feng, D.; Xu, L.; Pan, X.; Wang, X. Jamming wideband radar using interrupted-sampling repeater. *IEEE Trans. Aerosp. Electron. Syst.* **2017**, *53*, 1341–1354. [CrossRef]
3. Huang, G.; Yang, L. A radar anti-jamming technology based on blind source separation. In Proceedings of the 7th International Conference on Signal Processing, Beijing, China, 31 August–4 September 2004.
4. Greco, M.; Gini, F.; Farina, A. Radar detection and classification of jamming signal belonging to a cone class. *IEEE Trans. Signal Process.* **2008**, *56*, 1984–1993. [CrossRef]
5. Bandiera, F.; Farina, A.; Orlando, D.; Ricci, G. Detection algorithms to discriminate between radar targets and ECM signal. *Trans. Signal Process.* **2010**, *58*, 5985–5993. [CrossRef]
6. Zhao, S.; Zhou, Y.; Zhang, L.; Tang, S. Discrimination between radar targets and deception jamming in distributed multiple-radar architectures. *IET Radar Sonar Navig.* **2011**, *5*, 12–22. [CrossRef]
7. Hao, Z.; Yu, W.; Chen, W. Recognition method of dense false targets jamming based on time-frequency atomic decomposition. *J. Eng.* **2019**, *2019*, 6354–6358. [CrossRef]
8. Liu, Y.; Wang, W.; Pan, X.; Dai, D.; Feng, D. A-frequency-domain three-stage algorithm for active deception jamming against synthetic aperture radar. *IET Radar Sonar Navig.* **2014**, *8*, 639–646. [CrossRef]
9. Ma, X.; Qin, J.; Li, J. Pattern recognition-based method for radar anti-deceptive jamming. *J. Syst. Eng. Electron.* **2005**, *4*, 802–805.
10. Tian, X.; Tang, B.; Gui, G. Product spectrum matrix feature extraction and recognition of radar deception jamming. *Int. J. Electron.* **2013**, *100*, 1621–1629. [CrossRef]
11. Liu, Y.; Xing, S.; Li, Y.; Hou, D.; Wang, X. Jamming recognition method based on the polarization scattering characteristics of chaff clouds. *IET Radar Sonar Navig.* **2017**, *11*, 1689–1699. [CrossRef]

12. Qin, F.; Meng, J.; Du, J.; Ao, F.; Zhou, Y. Radar jamming effect evaluation based on AdaBoost combined classification model. In Proceedings of the 2013 IEEE 4th International Conference on Software Engineering and Service Science, Beijing, China, 23–25 May 2013; pp. 910–913.
13. Gao, M.; Li, H.; Jiao, B.; Hong, Y. Simulation research on classification and identification of typical active jamming against LFM radar. In Proceedings of the 11th International Conference on Signal Processing Systems. International Society for Optics and Photonics, Chengdu, China, 31 December 2019.
14. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
15. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
16. Wang, Y.; Sun, B.; Wang, N. Recognition of radar active-jamming through convolutional neural networks. *IET J. Eng.* **2019**, *2019*, 7695–7697. [CrossRef]
17. Liu, Q.; Zhang, W. Deep learning and recognition of radar jamming based on CNN. In Proceedings of the 2019 12th International Symposium on Computational Intelligence and Design, Hangzhou, China, 14–15 December 2019; pp. 208–212.
18. Qu, Q.; Wei, S.; Liu, S.; Liang, J. Jamming recognition networks for radar compound suppression jamming signal. *IEEE Trans. Veh. Technol.* **2020**, *69*, 15035–15045. [CrossRef]
19. Wu, Z.; Zhao, Y.; Yin, Z.; Luo, H. Jamming signal classification using convolutional neural network. In Proceedings of the 2017 IEEE International Symposium on Signal Processing and Information Technology, Bilbao, Spain, 18–20 December 2017; pp. 62–67.
20. Andriyanov, N.; Dementiev, V.; Gladkikh, A. Analysis of the Pattern Recognition Efficiency on Non-Optical Images. In Proceedings of the 2021 IEEE Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT), Online, 13–14 May 2021; pp. 319–323.
21. Shao, G.; Chen, Y.; Wei, Y. Convolutional neural network-based radar jamming signal classification with sufficient and limited samples. *IEEE Access* **2020**, *8*, 80588–80598. [CrossRef]
22. Nikias, C.L.; Raghuveer, M.R. Bispectrum estimation: A digital signal processing framework. *Proc. IEEE.* **1987**, *75*, 869–891. [CrossRef]
23. Li, J.; Shen, Q.; Yan, H. Signal feature analysis and experimental verification of radar deception jamming. In Proceedings of the 2011 IEEE CIE International Conference on Radar, Chengdu, China, 24–27 October 2011; pp. 230–233.
24. Zhou, H.; Dong, C.; Wu, R.; Xu, X.; Guo, Z. Feature fusion based on Bayesian decision theory for radar deception jamming recognition. *IEEE Access* **2021**, *9*, 16296–16304. [CrossRef]
25. Cheng, Y.; Wang, D.; Zhou, P.; Zhang, T. A Survey of model compression and acceleration for deep neural networks. *arXiv* **2017**, arXiv:1710.09282.
26. He, Y.; Zhang, X.; Sun, J. Channel pruning for accelerating very deep neural networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
27. He, Y.; Liu, P.; Wang, Z.; Hu, Z.; Yang, Y. Filter pruning via geometric median for deep convolutional neural networks acceleration. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 4340–4349.
28. Wang, W.; Fu, C.; Guo, J.; Cai, D.; He, X. COP: Customized deep model compression via regularized correlation-based filter-level pruning. *arXiv* **2019**, arXiv:1906.10337.
29. Wu, Q.; Zhao, F.; Ai, X.; Liu, X.; Xiao, S. Two-dimensional blanket jamming against ISAR using nonperiodic ISRJ. *IEEE Sens. J.* **2019**, *19*, 4031–4038. [CrossRef]
30. Shi, R.; Xu, J. Multipath effect analysis and pre-distortion processing for jamming on wideband ground radar through antenna sidelobe. *IET J. Eng.* **2019**, *2019*, 5672–5676. [CrossRef]
31. Wen, C.; Peng, J.; Zhou, Y.; Wu, J. Enhanced three-dimensional joint domain localized STAP for airborne FDA-MIMO radar under dense false-target jamming scenario. *IEEE Sens. J.* **2018**, *18*, 4154–4166. [CrossRef]
32. Biau, G.; Scornet, E. A random forest guided tour. *Test* **2016**, *25*, 197–227. [CrossRef]
33. Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.* **2002**, *21*, 660–674.
34. Altman, N.S. An introduction to kernel and nearest-neighbor nonparametric regression. *Am. Stat.* **1992**, *46*, 175–185.