Progress In Electromagnetics Research C, Vol. 126, 183–196, 2022

Network Optimization Algorithm for Radar Active Jamming Identification Based on Neural Architecture Search

Zejun Gao, Fei Cao*, Chuan He, Xiaowei Feng, Jianfeng Xu, and Jianqiang Qin

Abstract—A network optimization approach based on Neural Architecture Search (NAS) and network pruning is suggested to solve the issue of poor recognition performance of missile-borne radar active jamming under the condition of short sample sizes. The approach realized the ideal network design under severely constrained technical indications by combining the benefits of several methods, including NAS, convolutional light-weighting, and network pruning. The recognition network's convolution kernel size parameters were first optimized using NAS. The number of model parameters were then decreased via convolutional substitution. Finally, the structured pruning algorithm further screened the redundant network based on the technical indicators. The WideResNet28_2 wide residual network's recognition accuracy is only 84.38% when there are only 1000 training samples for each type of signal, according to the simulation results. After optimization, the number of new model parameters was increased to 2.55 M, 2.26 M, and 1.78 M, respectively, and their respective recognition accuracy was increased to 85.7%, 85.61%, and 85.37%. According to the simulation results, the technique offers a wide range of possible applications in the optimized design of radar active jamming identification networks for small sample sizes.

1. INTRODUCTION

Active spoofing jamming with high fidelity and high intelligence are emerging with the rapid development of technologies like solid-state circuits, very large scale integrated circuits (VLSI), and digital radio frequency memory (DRFM), posing a serious danger to radar systems [1]. To guarantee that radar operates well and can survive in challenging electromagnetic environments, it is essential to identify the different forms of active jamming.

Among deep learning models, convolutional neural network (CNN) has had considerable success in many different disciplines and can automatically learn discriminative and invariant features from data [2–4]. CNN has also had some success, particularly in the area of radar interference type recognition [5, 6]. The development of deep neural networks has shown that adding more neural network layers can enhance a model's capacity for generalization. More layers, however, do not always equate higher performance [7]. One factor that makes deep learning successful is "a large amount of data". Google has investigated how "large amount of data" and deep learning are related. It demonstrates unequivocally that the larger the dataset is, the more effective the performance is by conducting deep learning trials with 300 million photos [8]. However, in radar work scenarios, especially in the face of electronic countermeasures, the difficulty of obtaining data increases considerably, leading to a large randomness and uncertainty of the number of samples used in the training process [9–11]. Therefore, the fundamental approach to enhancing radar performance is to optimize the network in accordance with the application scenario.

Deep neural networks can now be built automatically using algorithms, outperforming traditional hand-designed networks in terms of performance [12–14]. A lightweight network based on neural network

Received 18 August 2022, Accepted 1 November 2022, Scheduled 10 November 2022
* Corresponding author: Fei Cao (lyakhp@163.com).
The authors are with the Xi'an Research Institute of High Technology, Xi'an 710025, China.

structure search was proposed by Yao et al. [13] to address the grouping structure problem in the grouping convolution that is frequently employed in lightweight networks. By phrasing tasks in a differentiable way, Liu et al. [14] addressed the scalability issue of architecture search. Specifically, they converted the network architecture search problem into a comparable network pruning problem based on a differentiable architecture search approach. The fundamental purpose of network pruning is to eliminate duplicate structures and parameters from deep learning models while maintaining the model's ability to make accurate predictions [15–18]. A CNN acceleration technique was put forth by Li et al. [16], which excluded filters thought to have little bearing on the network's output accuracy. A fresh channel pruning technique was put out by He et al. [17] to speed up deep convolutional neural networks. A quick soft filter pruning technique was put out by He et al. [18] for the deep convolutional neural network inference process. In lightweight neural networks, convolution is a crucial element [19]. The size of the network receptive field is determined by the choice of the convolution kernel size. The receptive field increases with the size of the convolution kernel [20]. However, the experimental findings from numerous models have demonstrated that using big convolution kernels would lead to a number of issues, including an abrupt rise in computational load, poor performance, and the inability to easily alter the depth [19]. So, for neural network optimization, modifying the convolution to achieve network lightweight is the preferable option. A method for dividing big size convolution kernels was presented by Szegedy et al. [21] based on an analysis and comparison of the computational effort needed for various size convolution kernels. This method makes use of the increased computation as effectively as feasible through the use of appropriate factorial convolution and aggressive regularization. However, there are few reports of the aforementioned study being used to identify the type of active jamming in radar systems.

This research suggests a network optimization approach for radar active jamming identification based on NAS and network pruning to solve the aforementioned issues. The time-frequency analysis theory, active jamming model, and radar echo are first briefly introduced. Second, the network optimization algorithm based on NAS and network pruning for identifying radar active jamming is examined. Finally, simulation verification of the type identification of radar active jamming is performed using limited samples.

2. THEORY ON RADAR ACTIVE JAMMING SIGNAL MODELING AND TIME-FREQUENCY ANALYSIS

This section first introduces the pseudo random code pulse Doppler radar target echo and active jamming signal model and time-frequency analysis theory, followed by model simulation with reasonable parameters, and finally introduces the dataset structure used in this paper.

2.1. Radar Echo and Active Jamming Signal Model

Because finding publicly available datasets for radar active jamming research is challenging, and building and implementing real-world test environments for electronic countermeasures is expensive, modeling and simulation are key tools for conducting studies on the design of radar systems. The radar target echo and active jamming signal model in this paper are shown in Table 1 [22–24].

2.2. Time-Frequency Analysis Theory

The signal is first preprocessed by time-frequency analysis using the transform domain, where the high-dimensional features that are retrieved can better characterize the signals with various modulation patterns [25, 26]. The pseudo-Wigner-Ville distribution (PWVD) for nonlinear and non-stationary radar signals has high time-frequency aggregation and good cross-term suppression effect, and its calculation is also very straightforward [27]. Therefore, in this work, the properties of radar active jamming signals are extracted using the pseudo-Wigner-Ville distribution. PWVD is described as follows [28]:

$$PWD\left(t,\omega\right)=\int_{-\infty}^{\infty}x\left(t+\frac{\tau}{2}\right)x^{\star}\left(t-\frac{\tau}{2}\right)h\left(\tau\right)e^{-j\omega\tau}d\tau. \tag{1}$$

Table 1. The radar target echo and active jamming signal models.

Signal types	Formula	Explanation
Radar target echo [23]	$S_r(t) = Rect_A\left[\frac{t-\tau_R}{\tau_A}\right] A_r P\left(t-\tau_R\right)\cos\left[\omega_0\left(t-\tau_R\right)\right]$ $= Rect_A\left[\frac{t-\tau_R}{\tau_A}\right] A_r P\left(t-\tau_R\right)\cos\left(\omega_0 t+\omega_d t\right)$	A_r: echo signal amplitude; τ_R: projectile distance delay; c: light speed; ω_d: Doppler angle frequency.
Noise AM jamming [23]	$J_A(t) = [A_0 + n(t)]\cos\omega_j t$	A_0: carrier amplitude; ω_j: carrier angular frequency; $n(t)$: generalized stationary noise with mean 0 and variance σ^2.
Noise FM jamming [23]	$J\left(t\right) = U_j\cos\left[2\pi f_j t + 2\pi K_{FM}\int_0^t u_n\left(t'\right)dt' + \phi\right]$	U_j: amplitude of the modulated signal; f_j: carrier frequency of the modulated signal; $u_n(t)$: generalized stationary random process with zero mean and variance σ^2, i.e., modulated noise.
Noise PM jamming [23]	$J\left(t\right) = U_j\cos\left[2\pi f_j t + K_{PM}u_n\left(t\right) + \phi\right]$	U_j: amplitude of the modulated signal; f_j: carrier frequency of the modulated signal; K_{FM}: phase modulation slope of a noisy phase modulated signal; $u_n(t)$: generalized stationary random process with zero mean and variance σ^2, i.e., modulated noise; ϕ: The initial phase is uniformly distributed on $[0, 2\pi]$, and satisfies the independent distribution relationship with $u_n(t)$.
Range deception jamming [23]	$S_j(t) = Rect_A\left[\frac{t-\tau_j}{\tau_A}\right] A_j P\left(t-\tau_j\right)\cos\left(\omega_0(t-\tau_j)\right)$	A_j: amplitude of the interfering signal received by the receiver; τ_j: total jamming delay, including jammer inherent delay and jamming delay set by jammer; $P(t)$: The same pseudo-random code signal as the target signal.
Intermittent sampling direct forwarding interference [24]	$S_J(t) = \sum_{n=0}^{N-1} rect\left(\frac{t-\tau-(2n+1)T_J}{T_J}\right)$ $\cdot\exp\left[j\pi K_r\left(t-\tau-T_J\right)^2\right]$	$rect(\cdot)$: wave gate function; N: number of interference slices; T_J: width of interference slice; K_r: transmit signal frequency; τ: jammer-to-radar distance induced delay.
Intermittent sampling and repeated forwarding interference [24]	$S_J(t) = \sum_{m=1}^{M}\sum_{n=0}^{N-1} rect\left(\frac{t-\tau-nT_n-mT_J}{T_J}\right)$ $\cdot\exp\left[j\pi K_r\left(t-\tau-mT_J\right)^2\right]$	M: The number of times each slice was forwarded; $T_u = (M+1)\cdot T_J$: The time interval at which the jammer intercepts the signal.

2.3. Parameter Settings

2.3.1. Pseudo-Random Code Pulse Doppler Radar Echo

Parameter settings: the pseudo-random code symbol width $T_c = 50\,\text{ns}$, code length $P = 31$, so the pulse width $T = 1\,\mu\text{s}$, the pulse repetition period $T_R = 5\,\mu\text{s}$, the carrier frequency $f_0 = 220\,\text{MHz}$, the bullet distance $R_t = 60\,\text{m}$, and the signal-to-noise ratio is $10\,\text{dB}$. If there is not any interference, the target

echo signal is processed to create an intermediate frequency signal, which is then output after being correlated with the signal from the local oscillator.

2.3.2. Radar Active Jamming Parameter Settings and Simulation Results

2.3.2.1 Parameter Settings

Suppressive jamming parameters: the interference signal ratio is 10 dB; the frequency modulation slope

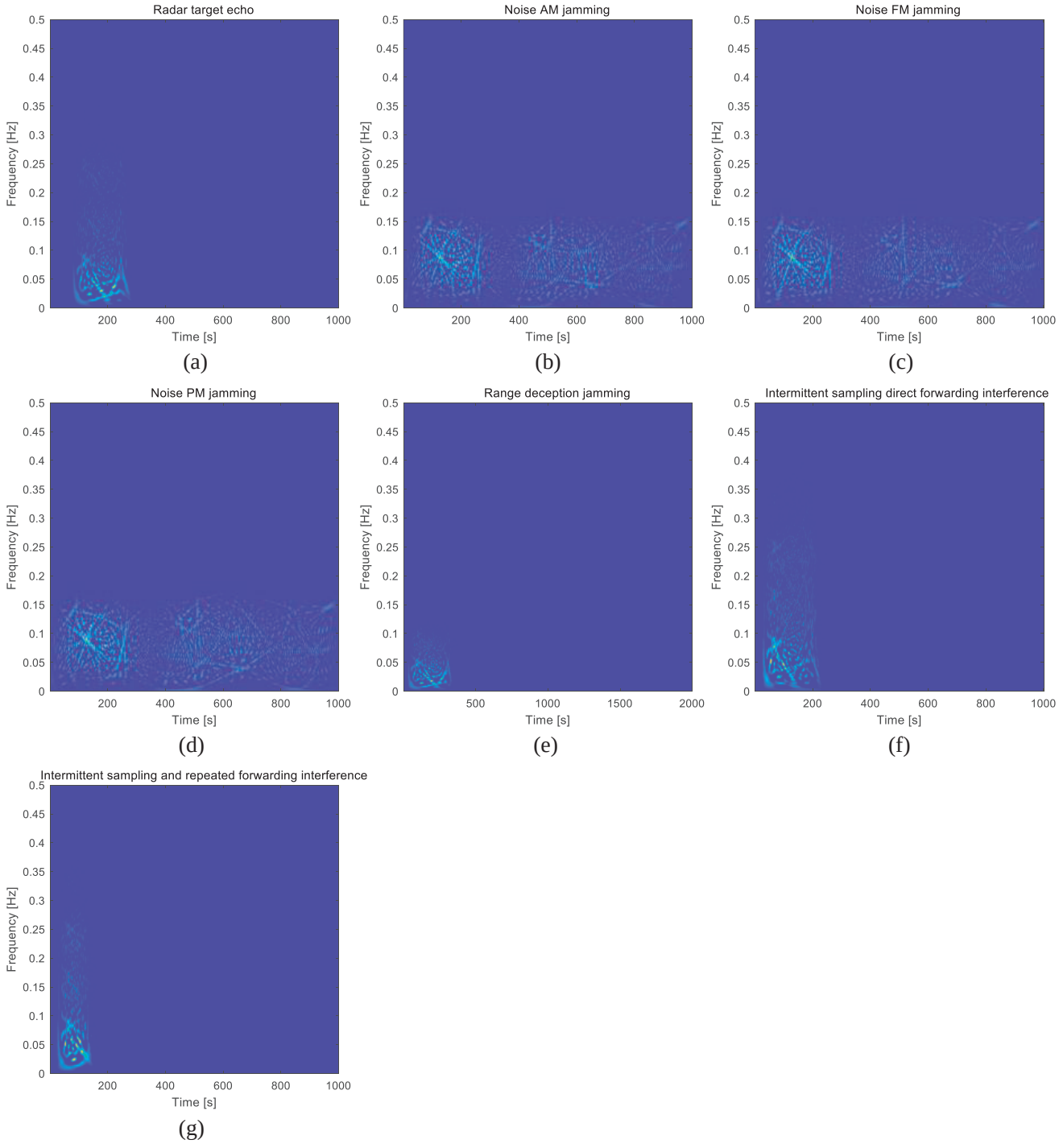

Figure 1. PWVD of radar target echo and active jamming.

is $10e^6$; the phase modulation slope is 3; and the signal bandwidth is 20 MHz. In order to realize the suppression interference, the suppression signal bandwidth is 80 MHz. The deception interference parameters: the interference signal ratio is 5 dB, and deception distance $R_j = 20$ m. Intermittent sampling forwarding jamming parameters: the signal-to-interference ratio is 13 dB, and deception distance $R_j = 20$ m. The rest of the parameters remain unchanged as shown in Section 2.3.1.

2.3.2.2 Simulation Environment

This part is implemented using MATLAB 2021b; the CPU is 1 Intel(R) Core(TM) i7-10750H processor; and the GPU is c 1 NVIDIA GeForceGTX2060 graphics card.

2.3.2.3 Simulation Results

This section is based on Matlab simulation. The simulated time-frequency distribution diagram of the intermediate frequency target echo and interference signal after mixing are shown in Figure 1.

2.4. Dataset Construction

2.4.1. Dataset Structure

The CIFAR10 [29] dataset, which contains 1 type of radar target echo and 6 types of active interference, is used to generate the dataset in this research. The training set, test set, and class labels make up the dataset's three components. The training set is separated into 5 batches, while the test set contains just 1 batch, with each batch storing 7000 three-channel grayscale images ($32 \times 32 \times 3$) and a total of 42,000 grayscale images; the class labels for each sample are recorded in the file batches. meta.bet (As shown in Table 2).

Table 2. The radar target echo and active jamming type labels.

Signal types	Label
Radar target echo	real-echo
Noise AM jamming	b-pam
Noise FM jamming	b-pfm
Noise PM jamming	b-ppm
Range deception jamming	d-range
Intermittent sampling direct forwarding interference	i-direct
Intermittent sampling and repeated forwarding interference	i-repeater

2.4.2. Parameter Settings of Radar Active Jamming Model

Signal samples are collected based on MATLAB simulation environment: set the value range of interference signal ratio (ISR) to $10 \sim 50$ dB and the value range of SNR to $-10 \sim 10$ dB.

3. PROPOSED ALGORITHM

Motivation: The size of a model's receptive field has a direct impact on how well it performs, with the convolution kernel's size being the primary determinant of receptive field size. The convolution kernel size can be increased to enhance the receptive field, although doing so increases model parameters by an order of magnitude. The model's performance can be enhanced while limiting the growth of model parameters by using the convolution replacement method.

Figure 2 depicts the network optimization algorithm for identifying radar active jamming based on NAS and network pruning.

The specific steps of the algorithm are as follows:

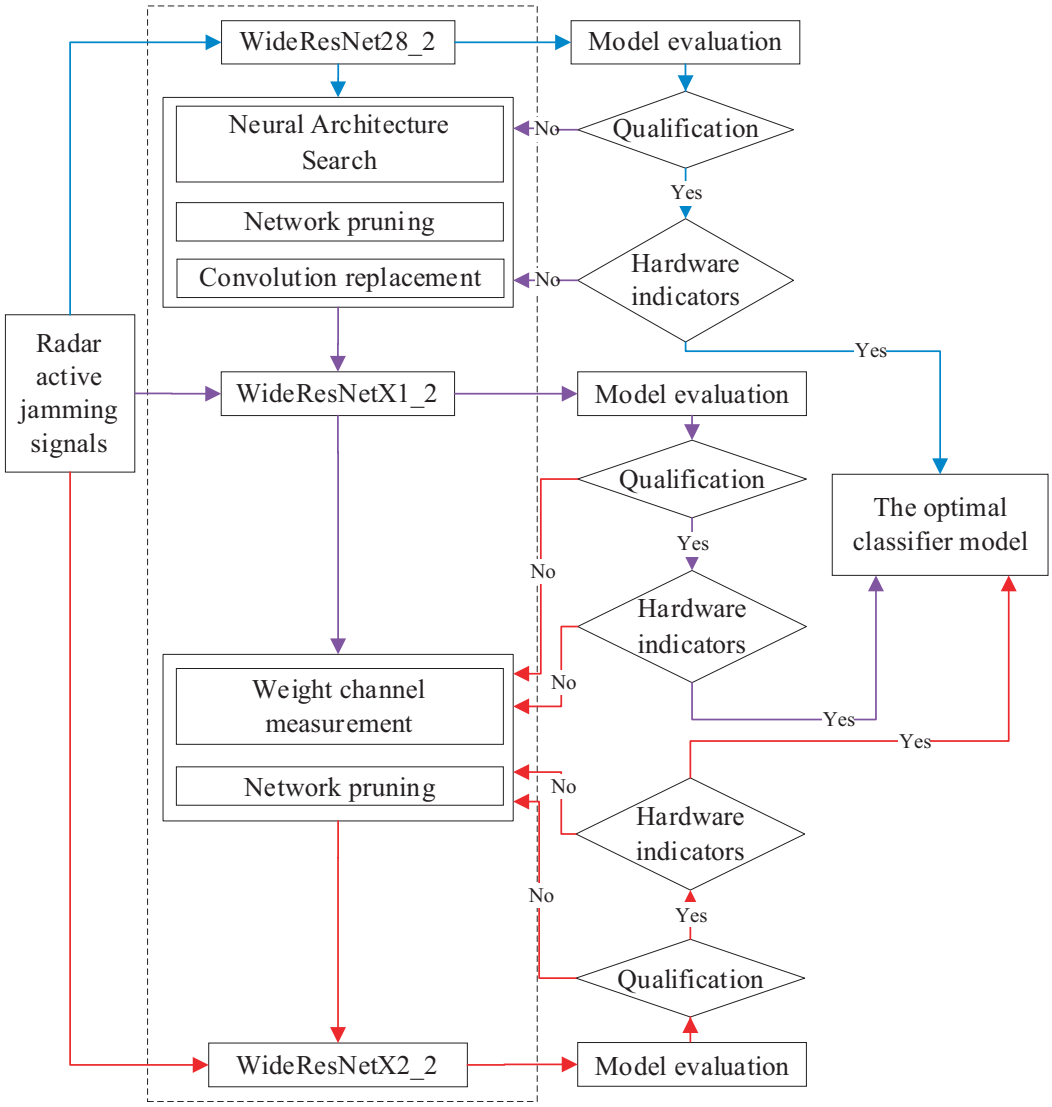

Figure 2. Flow chart of the proposed algorithm.

Step 1: (as shown by the blue arrow) Perform model evaluation on the initial model (residual network WideResNet28 2 [30, 31]) before analyzing the radar active interference dataset. It is the best classifier model if the performance satisfies the hardware specifications and index requirements; else, move on to the next step.

Step 2: (as shown by the purple arrow) The network convolution kernel parameters should be first optimized using neural network structure search and network pruning. The WideResNet28_2 network's convolution kernel size parameters are divided into five categories: $(3, 3)$, $(5, 5)$, $(7, 7)$, $(9, 9)$, and $(11, 11)$. Next, utilize convolution replacement to get the best classifier (WideResNetX1_2) for the dataset by reducing the model parameters. Then evaluate the model using it. It is the best classifier model if it satisfies the aforementioned criteria; otherwise, move on to step three.

Step 3: (as shown by the red arrow) The best classifier model (WideResNetX2 2) was obtained in this application scenario by measuring and sorting the weight channels of the new model and then performing network pruning in accordance with the requirements and constraints to significantly increase the generalization ability of the model.

4. SIMULATION RESULTS AND ANALYSIS

The simulation test primarily looks at how the model's performance is impacted by a small sample size. In the experiment, three simulated application situations are built up, and the performance indicators are primarily analyzed using a confusion matrix and recognition accuracy rate.

4.1. Simulation Environment and Experimental Settings

4.1.1. Simulation Environment and Experimental Setup

4.1.1.1 Simulation Environment

The experiment was carried out using the Python programming language and the free and open-source Pytorch0.4 deep learning framework. An NVIDIA GeForce GTX3080 graphics card was used as the GPU, which assisted the CPU set up with an Intel(R) Core(TM) i9-11900 processor. On the CIFAR-Radar7 dataset, the algorithm's optimization effect was confirmed.

4.1.1.2 Experimental Setup

In this experiment, three application scenarios were built up to assess the algorithm's performance in accordance with various requirements, such as technical indicators and hardware configuration.

(1) Application scenario 1
- ① Dataset: the number of signal samples of each type in the training set is divided into three cases: 5000, 3000, and 1000;
- ② Technical indicators: no special requirements;
- ③ Hardware configuration: no special requirements;
- ④ Experiment content: evaluate the performance of the WideResNet28_2 classifier with different numbers of training sets.

(2) Application scenario 2
- ① Dataset: the number of samples of each type in the training set is 1000;
- ② Technical indicators: recognition accuracy rate $\geq 85.5\%$;
- ③ Hardware configuration: model parameters ≤ 2.8 M;
- ④ Experiment content: evolution of the best parameters of the model under small sample conditions; based on WideResNet28_2 classifier, the model is optimized according to conditions ② and ③, and its performance is tested.

(3) Application scenario 3
- ① Dataset: the number of samples of each type in the training set is 1000;
- ② Technical indicators: recognition accuracy rate $\geq 85\%$;
- ③ Hardware configuration: model parameters ≤ 2.4 M; model parameters ≤ 2.0 M.
- ④ Test content: for the optimal model in scenario 2, optimize the design according to the conditions ② and ③ in this scenario.

4.2. Simulation Results

4.2.1. Evaluating the Performance of the WideResNet28_2 Classifier

4.2.1.1 Performance of the Trained Model for Signal Recognition with 5000 Training Samples

1.47 M was the model parameter. The recognition accuracy was 87.5% following 1200 training rounds. Figure 3 displays the results of the recognition.

4.2.1.2 Performance of the Trained Model at Recognizing Each Type of Signal Using 3000 Training Samples

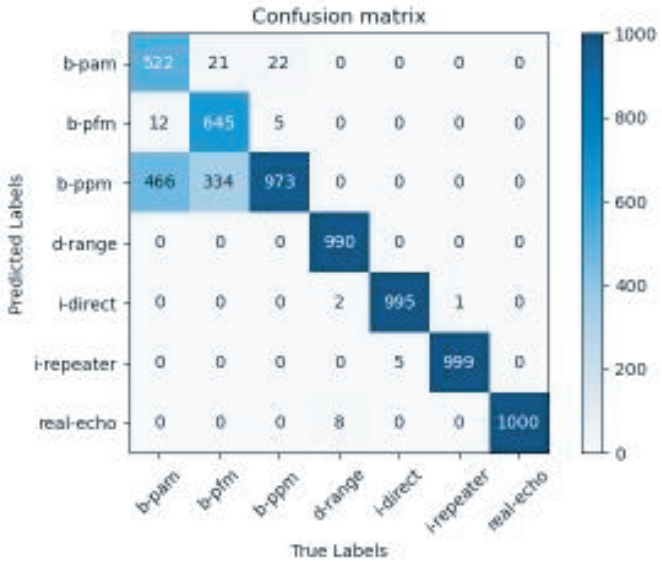

Figure 3. Confusion matrix.

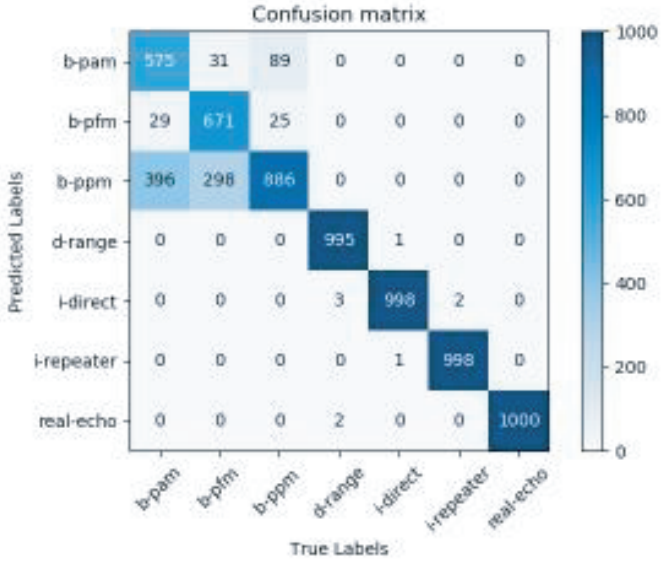

Figure 4. Confusion matrix.

The 1.47 M model parameter. The recognition accuracy rate was 87.47% after 1200 training rounds. Figure 4 displays the recognition outcomes.

4.2.1.3 Performance of the Trained Model for Each Type of Signal with 1000 Training Samples

1.47 M was the model parameter. It took 1200 training cycles to achieve an accuracy rate of 84.38% in recognition. Figure 5 displays the recognition results.

4.2.2. Optimizing the Design of the WidResNet28_2 Classifier

The WideResNet28 2 network's model parameters were optimized using NAS under the restriction of just 1000 training samples for each type of signal. The parameters of the new model NewWideResNet28_2 are displayed in Table 3 following training.

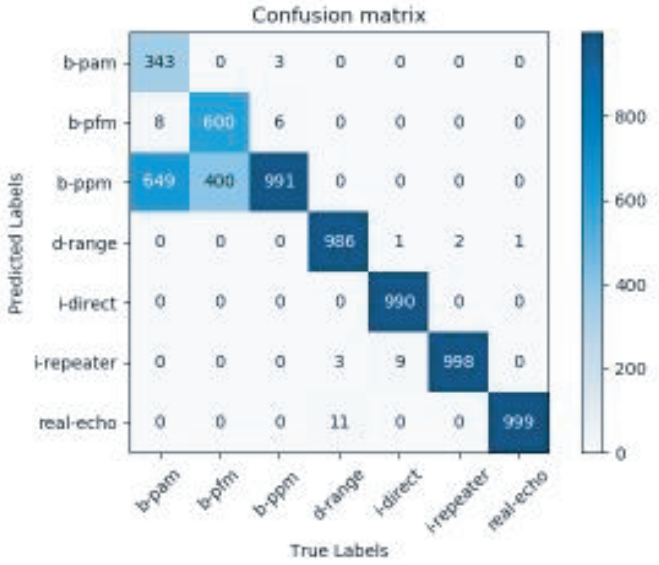

Figure 5. Confusion matrix.

Convolution replacement was used to lighten the NewWideResNet28_2 model, which led to the creation of the NewWideResNet50_2 model. Its primary attributes are displayed in Table 4.

The identification accuracy of NewWideResNet50_2 obtained 85.7% after 1200 training rounds under the assumption that the training set contained 1000 samples. Figure 6 displays the recognition results.

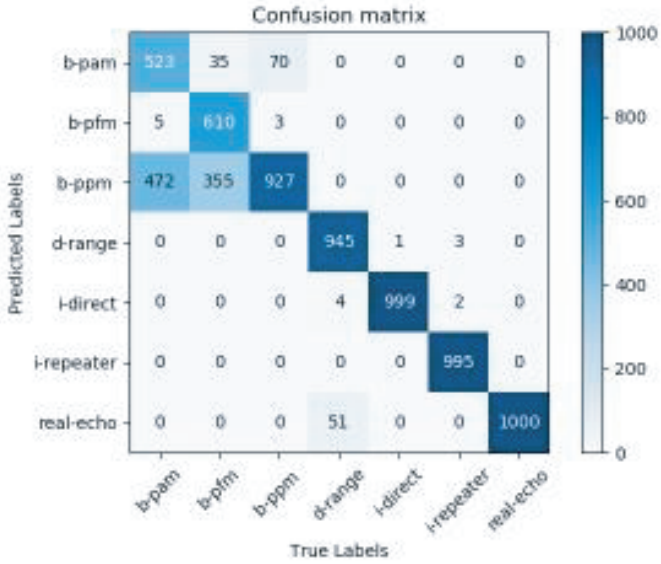

Figure 6. Confusion matrix.

4.2.3. Optimize and Evaluate New Models according to Technical Indicators

The algorithm was applied to NewWideResNet50_2 optimization. Table 5 displays the key characteristics of the ideal classifier model under various usage scenarios.

The recognition outcomes of NewWideResNet48_2 are displayed in Figure 7 after 1200 training rounds under the assumption that the training set contained 1000 samples. Figure 8 displays the NewWideResNet38_2 recognition outcomes.

Table 3. The main parameters of the WideResNet28_2 model.

		Convolution Layer	Convolution Kernel	Input Size	output Size
Input	conv1		(3, 3)	3	16
Block1	BasicBlock(0)	conv1	(3, 3)	16	32
		conv2	(3, 3)	32	32
		short	(1, 1)	16	32
	BasicBlock (1)	conv1	(3, 3)	32	32
		conv2	(3, 3)	32	32
	BasicBlock (2)	conv1	(7, 7)	32	32
		conv2	(7, 7)	32	32
	BasicBlock (3)	conv1	(5, 5)	32	32
		conv2	(5, 5)	32	32
Block2	BasicBlock (0)	conv1	(3, 3)	32	64
		conv2	(3, 3)	64	64
		short	(1, 1)	32	64
	BasicBlock (1)	conv1	(7, 7)	64	64
		conv2	(7, 7)	64	64
	BasicBlock (2)	conv1	(7, 7)	64	64
		conv2	(7, 7)	64	64
	BasicBlock (3)	conv1	(7, 7)	64	64
		conv2	(7, 7)	64	64
Block3	BasicBlock (0)	conv1	(5, 5)	64	128
		conv2	(5, 5)	128	128
		short	(1, 1)	64	128
	BasicBlock (1)	conv1	(5, 5)	128	128
		conv2	(5, 5)	128	128
	BasicBlock (2)	conv1	(3, 3)	128	128
		conv2	(3, 3)	128	128
	BasicBlock (3)	conv1	(3, 3)	128	128
		conv2	(3, 3)	128	128
Output	fc			128	7
#Params				3330048	

4.3. Discussion

According to the above simulation results, the analysis is as follows:

(1) As depicted in Figures 3, 4, and 5, and as demonstrated by the change in accuracy, the model's performance declines dramatically when the number of samples is drastically reduced.

(2) The ideal classifier receptive field parameters have been altered, as shown in Table 3, when few samples are present. As seen in Tables 4 and 3, the number of model parameters are lowered by 23.4% after convolution replacement. As seen in Figures 6, 7, and 8, the classifier model successfully raises the model's recognition rate while minimally raising the model's parameters, which satisfies the demands of the technical indicators and hardware configuration.

(3) As seen in Table 5, the technique may be used to more tightly tune the classifier model so that both the model parameters and recognition rate fulfill the requirements of the indicators for

more stringent technical indicators and hardware configuration requirements. The device's capacity for generalization is significantly enhanced.

Table 4. The main parameters of the WideResNet50_2 model.

		Convolution Layer	Convolution Kernel	Input Size	output Size
Input	conv1		(3, 3)	3	16
Block1	BasicBlock (0)	conv1	(3, 3)	16	32
		conv2	(3, 3)	32	32
		convShortcut	(1, 1)	16	32
	BasicBlock (1)	conv1	(3, 3)	32	32
		conv2	(3, 3)	32	32
	...				
	BasicBlock (6)	conv1	(3, 3)	32	32
		conv2	(3, 3)	32	32
Block2	BasicBlock (0)	conv1	(3, 3)	32	64
		conv2	(3, 3)	64	64
		convShortcut	(1, 1)	32	64
	BasicBlock (1)	conv1	(3, 3)	64	64
		conv2	(3, 3)	64	64
	...				
	BasicBlock (9)	conv1	(3, 3)	64	64
		conv2	(3, 3)	64	64
Block3	BasicBlock (0)	conv1	(3, 3)	64	128
		conv2	(3, 3)	128	128
		convShortcut	(1, 1)	64	128
	BasicBlock (1)	conv1	(3, 3)	128	128
		conv2	(3, 3)	128	128
	...				
	BasicBlock (5)	conv1	(3, 3)	128	128
		conv2	(3, 3)	128	128
Output	fc			128	7
#Params			2551088		

Table 5. The main parameters of the optimal classifier model.

Network Structure		WideResNet38_2	WideResNet48_2	WideResNet50_2
Input	conv1	1	1	1
Block1	Basicblock	5	7	7
Block2	Basicblock	8	10	10
Block3	Basicblock	4	5	6
Output	fc	1	1	1
#Params		1.78 M	2.26 M	2.55 M
Accuracy		85.37%	85.61%	85.7%

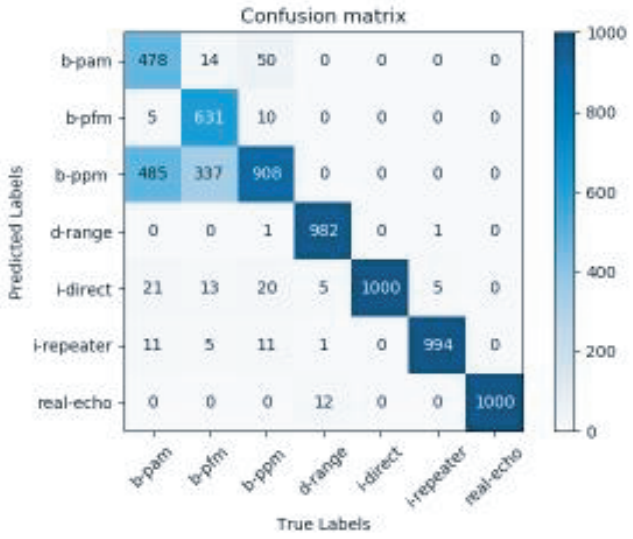

Figure 7. Confusion matrix.

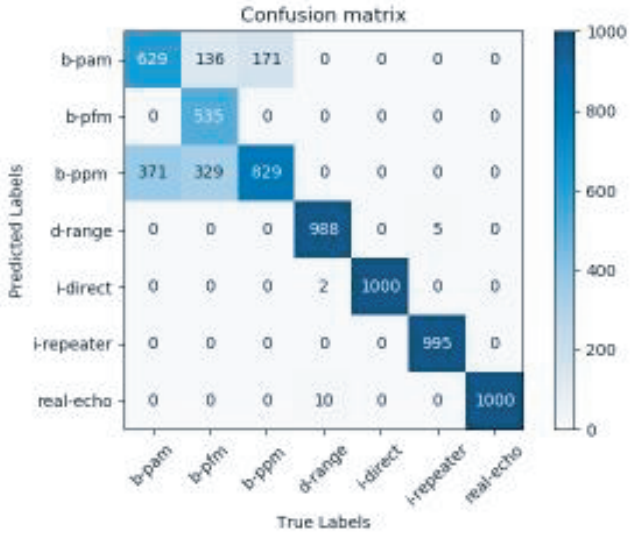

Figure 8. Confusion matrix.

In conclusion, increasing the receptive field is the fundamental strategy for enhancing the model's performance in the case of small samples. The model's receptive field can be enhanced by increasing the size of the convolution kernel; however, this will result in a data-level rise in the model parameters. Using convolution replacement can change the convolution kernel size optimization problem into a network depth optimization problem, so enhancing its performance while successfully controlling the increase in model parameters.

5. CONCLUSION

This research proposes a network optimization approach based on NAS and network pruning for identifying radar active jamming. Utilizing technologies like as NAS, convolution replacement, and network pruning, the algorithm realizes the ideal network design for a variety of application scenarios. The simulation results shown that the algorithm efficiently enhances the operational performance of the radar system when the technical indicators of the application scene change, with a high degree of generalizability. This provides an important method reference and data support for the dynamic optimization of the model in the interaction with application scenarios.

ACKNOWLEDGMENT

The work was supported by the Youth Science Foundation of National Natural Science Foundation of China (Grant No. 61903375).

REFERENCES

1. Li, N. J. and Y. T. Zhang, "A survey of radar ECM and ECCM," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 31, No. 3, 1110–1120, 1995.
2. Krizhevsky, A., I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, Vol. 25, 2012.
3. Sun, J., G. Xu, W. Ren, et al., "Radar emitter classification based on unidimensional convolutional neural network," *Radar, Sonar & Navigation*, Vol. 12, No. 8, 862–867, IET, 2018.
4. Liu, W., Z. Wang, X. Liu, et al., "A survey of deep neural network architectures and their applications," *Neurocomputing*, Vol. 234, 11–26, Apr. 19, 2017.
5. Yu, J., J. Li, B. Sun, et al., "Barrage jamming detection and classification based on convolutional neural network for synthetic aperture radar," *IGARSS 2018 — 2018 IEEE International Geoscience and Remote Sensing Symposium*, 4583–4586, IEEE, 2018.
6. Liu, S. and C. Zhu, "Jamming recognition based on feature fusion and convolutional neural network," *Journal of Beijing Institute of Technology*, Vol. 31, No. 2, 9, 2022.
7. He, K., X. Zhang, S. Ren, et al., "Deep residual learning for image recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778, 2016.
8. Sun, C., A. Shrivastava, S. Singh, et al., "Revisiting unreasonable effectiveness of data in deep learning era," *Proceedings of the IEEE International Conference on Computer Vision*, 843–852, 2017.
9. Saptarshi, M. and J. Maiti, "Risk analysis using FMEA: Fuzzy similarity value and possibility theory based approach," *Expert Systems with Applications*, Vol. 41, No. 7, 3527–3537, 2014.
10. Versaci, M., G. Angiulli, P. Crucitti, et al., "A Fuzzy similarity-based approach to classify numerically simulated and experimentally detected carbon fiber-reinforced polymer plate defects," *Sensors*, Vol. 22, 4232, 2022.
11. Zadeh, L. A., "A note on similarity-based definitions of possibility and probability," *Information Sciences*, Vol. 267, 334–336, 2014.
12. Termritthikun, C., Y. Jamtsho, J. Ieamsaard, et al., "EEEA-Net: An early exit evolutionary neural architecture search," *Engineering Applications of Artificial Intelligence*, Vol. 104, No. 2, 104397, 2021.
13. Yao, X., Y. Shi, G. Huo, et al., "Lightweight model construction based on neural architecture search," *Pattern Recognition and Artificial Intelligence*, Vol. 34, No. 11, 1038–1048, 2021.
14. Liu, H., K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," arXiv preprint arXiv:1806.09055, 2018.
15. Reed, R. D., "Pruning algorithms — A survey," *IEEE Transactions on Neural Networks*, 1993.
16. Li, H., A. Kadav, I. Durdanovic, et al., "Pruning filters for efficient convnets," arXiv preprint arXiv:1608.08710, 2016.
17. He, Y., X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," *Proceedings of the IEEE International Conference on Computer Vision*, 1389–1397, 2017.
18. He, Y., G. Kang, X. Dong, et al., "Soft filter pruning for accelerating deep convolutional neural networks," arXiv preprint arXiv:1808.06866, 2018.
19. Ma, J., Y. Zhang, Z. Ma, et al., "Research progress of lightweight neural network convolution design," *Journal of Frontiers of Computer Science and Technology*, Vol. 16, No. 3, 17, 2022.
20. Dumoulin, V. and F. Visin, "A guide to convolution arithmetic for deep learning," arXiv preprint arXiv:1603.07285, 2016.

21. Szegedy, C., V. Vanhoucke, S. Ioffe, et al., "Rethinking the inception architecture for computer vision," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826, 2016.
22. Wang, X., J.-C. Liu, W. Zhang, et al., "The mathematical principle of intermittent sampling and forwarding interference," *Science in China (Series E)*, Vol. 36, 891–901, 2006.
23. Suo, W., "Research on jamming against pseudo-random code pulse-doppler fuse," Xidian University, Shaanxi, 2014.
24. Zhou, C., Q. Liu, and C. Hu, "Time-frequency analysis techniques for recognition and suppression of interrupted sampling repeater jamming," *Journal of Radars*, Vol. 8, 100–106, 2019.
25. Yu, H., X. Yan, R. Jia, et al., "Research on anti-jamming performance of M-sequence pseudo-random code phase modulation pulse doppler fuze," *ACTA Armamentarii*, Vol. 41, No. 03, 417–425, 2020.
26. Toole, J. M. O. and B. Boashash, "Fast and memory-efficient algorithms for computing quadratic time-frequency distributions," *Applied & Computational Harmonic Analysis*, Vol. 35, No. 2, 350–358, 2013.
27. Gao, Z., F. Cao, C. He, et al., "MATLAB simulation of performance evaluation model of time-frequency analysis method based on SVM," *2022 8th International Symposium on Sensors, Mechatronics and Automation System*, Suzhou, China, 2022.
28. Zhang, X., *Modern Signal Processing (Third Edition)*, Tsinghua University Press, Beijing, 2015.
29. Recht, B., R. Roelofs, L. Schmidt, et al., "Do cifar-10 classifiers generalize to cifar-10?," arXiv preprint arXiv:1806.00451, 2018.
30. Zagoruyko, S. and N. Komodakis, "Wide residual networks," arXiv preprint arXiv:1605.07146, 2016.
31. Sohn, K., D. Berthelot, N. Carlini, et al., "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *Advances in Neural Information Processing Systems*, Vol. 33, 596–608, 2020.