An Intelligent Denoising Method for Jamming Pattern Recognition under Noisy Conditions

Changhua YAO [1], Yang LI [1], Yufan CHEN [2], Kaixin CHENG [2]

[1] School of Electronic and Information Engineering, Nanjing University of Information Science and Technology, Ningliu Road 219, Nanjing Jiangsu, 210044, China
[2] College of Communication Engineering, Army Engineering University of PLA, Houbiaoying Road No. 8, Nanjing Jiangsu, 210007, China

ych2347@163.com, 18736728779@163.com, cyf@aeu.edu.cn, ckx_paper@outlook.com

Submitted October 27, 2023 / Accepted March 25, 2024 / Online first May 13, 2024

Abstract. *Accurate identification of jamming patterns is a crucial decision-making basis for anti-jamming in wireless communication systems. Current works still face challenges in fully considering the substantial influence of environmental noise on identification performance. To address the issue, this paper proposes an automatic threshold denoising-based deep learning model. The proposed method aims to mitigate the impact of noise on recognition performance within the feature space. Considering the challenges posed by nonlinear transformations in deep denoising, a shallow denoising approach based on deep learning is proposed. By constructing a dataset of 12 jamming patterns under noisy conditions, the proposed method exhibits excellent recognition performance and maintains a low computational cost.*

Keywords

Jamming pattern recognition, automatic threshold denoising, shallow layer denoising, convolutional neural network

1. Introduction

Malicious jamming seriously impacts the security and reliability of wireless communications. The accurate identification of jamming facilitates the implementation of suitable countermeasures to mitigate or suppress its impact, thereby minimizing the negative impact on communication quality. Traditional research in anti-jamming primarily focuses on signal processing, yet it encounters intricate challenges in real-world scenarios. Fortunately, the emergence of machine learning (ML) provides a promising solution for intelligent anti-jamming measures [1], [2]. Similarly, ML-based jamming identification has become an indispensable component of intelligent anti-jamming systems.

Existing ML-based research mainly focused on enhancing jamming pattern recognition performance under noise-free conditions [3–7]. Among them, methods based on spectral features have shown excellent performance. A sequence-based deep reinforcement learning (RL) algorithm without prior information is proposed, utilizing spectral features as input and employing convolutional neural networks (CNNs) to classify jamming patterns [6]. In dealing with open-set pattern recognition tasks where some jamming patterns lack training samples, a zero-shot learning-based jamming pattern recognition approach that utilizes spectrogram waterfall to characterize jamming features has demonstrated effective performance [7]. Most of the existing researches, conducted under high signal-to-noise ratio (SNR) conditions, do not fully consider the impact of noise on jamming pattern recognition. However, in practical applications, the influence of noise cannot be ignored. Jamming recognition under noisy conditions is still left to be explored. Consequently, the need for effective jamming recognition under noisy conditions has become increasingly evident, which is prompting further research.

In response to these challenges, there has been a growing research interest in ML-based automatic denoising algorithms recently. These approaches differ from conventional methods by incorporating threshold denoising techniques within deep neural networks (DNNs), enabling more efficient processing of input features by utilizing trainable thresholds. Introducing a denoising module into radio frequency (RF) signal recognition network could achieve automatic threshold denoising [8]. Inserting the soft threshold as a non-linear transformation layer into the deep architecture would enhance the feature extraction capability for noisy signals [9]. Adding denoising blocks in the middle layers of convolutional networks, followed by end-to-end adversarial training could eliminate noise features in the data [10]. In [11] preprocess the fringe pattern and the iterate the IR-CNN denoiser multiple times in its painting stage to identify the fringe pattern painting. Authors of [12] use resblocks for feature encoder firstly, then use U-Net for decomposition, and

DOI: 10.13164/re.2024.0322

finally use denoising module. A deep neural network based on denoising which is composed of multiple denoisers modules interleaved with back-projection modules that ensure the observation consistencies [13]. The above research mainly embeds denoising algorithms into each layer or deep layer of neural networks. However, this kind of approach brings significant computational overhead, which hinders the utility of the methods. To address this issue, this paper places the denoising module in the shallow layers of the network, allowing it to effectively handle noise in the data while maintaining a relatively low computational cost.

This paper proposes a novel deep learning-based denoising method for jamming pattern recognition. Meanwhile, explore the impact of embedding the denoising module at different depths of the neural network on recognition performance and propose a shallow-layer denoising algorithm. The main contributions of this paper are as follows:

- A deep learning-based shallow-layer denoising method is proposed. The proposed method effectively mitigates noise propagation, preventing its further impact on feature extraction and classification in the deep network layers.

- A dataset that involves twelve typical jamming patterns under various jamming-to-noise ratios (JNRs) is constructed. This dataset can be used to evaluate the performance of the proposed method in jamming pattern recognition under noisy conditions.

- The proposed method achieves better performance and lower computational cost. Extensive experimental validation confirms the effectiveness of the proposed method, demonstrating its capability for jamming pattern recognition tasks under noisy conditions.

2. System Model

The constructed system model in this paper is illustrated in Fig. 1. The network architecture encompasses convolutional blocks, threshold denoising modules, a flatten layer and fully connected layers. The convolutional block comprises convolutional layers, batch normalization layers and max pooling layers. Convolutional layers perform convolution operations by element-wise multiplication and summation between input feature maps and learned convolutional kernels, and this process extracts diverse-scale local features from input data. The max pooling layers select the maximum value within pooling regions as the output, reducing the size of the output feature maps. The flatten layers transform the feature maps into one-dimensional vectors. The proposed threshold denoising module comprises one-dimensional convolutional layers, fully connected layers and a soft threshold calculation module that can handle the impact on feature activations within the feature space.

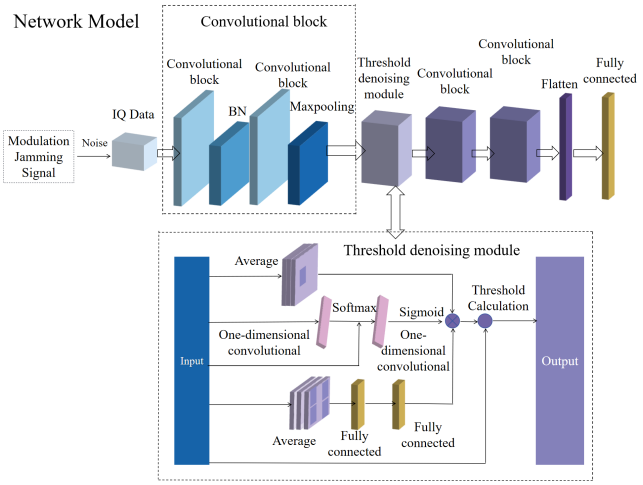


Fig. 1. System model.

As the number of parameters and layers in deep networks increases, they can learn more complex features. However, this heightened complexity also makes them more susceptible to the influence of noise in the data. Furthermore, deep networks introduce higher computational complexity and parameter counts than shallow networks, demanding more computational resources and time during training. Therefore, this paper introduces a shallow denoising approach, as illustrated in Fig. 1. A threshold denoising module is introduced after the first convolutional block, the primary objective is to minimize the model's parameter counts and computational complexity.

The reasonable selection of thresholds is a pivotal factor in achieving effective feature denoising. Many deep learning-based denoising models resort to global average pooling for threshold computation across each feature channel. However, distinct positions within feature maps encompass diverse information, and treating them uniformly with a single threshold yields suboptimal outcomes [8]. Therefore, this paper considers setting different thresholds for different positions on the feature map. When extracting a threshold for a specific position within the feature map, calculate the mean value of that position along the feature channel. Due to different feature channels having different importance to feature extraction, weighted averages are introduced to assign weights to different feature channels.

Taking the weighted average along the feature channels at specfic positions on the feature map can be expressed as follows:

$$\mathbf{Z}^a = \beta_1 g(\mathbf{x}_1^a) + \cdots + \beta_j g(\mathbf{x}_j^a) + \cdots + \beta_n g(\mathbf{x}_n^a) \quad (1)$$

where n represents the number of feature channels, $\mathbf{Z}^a$ denotes the weighted average value at different feature channels position a, β_j is the weight of channel j, $g(\mathbf{x}_j^a)$ is used to compute the mapping of input data at position a on the j-th feature channel, and $\mathbf{x}_j^a$ represents the feature value at the position a of the j-th feature channel, where $\beta_1 + \cdots + \beta_j + \cdots + \beta_n = 1$.

To compute the weights as described above, this paper employs the embedded Gaussian function to compute the weights, as described in [8]:

$$\beta_j = \sum_{j=1}^{n} \text{softmax}(\mathbf{W}_\text{k}\mathbf{x}_j). \tag{2}$$

Substituting (2) into (1) yields:

$$\mathbf{Z}^a = \sum_{j=1}^{n} \rho(\mathbf{W}_\text{k}\mathbf{x}_j) g\left(\mathbf{x}_j^a\right) \tag{3}$$

where $\rho(\cdot)$ represents the function of softmax, $g\left(\mathbf{x}_j^a\right) = \mathbf{W}_\text{v}\mathbf{x}_j^a$, $\mathbf{W}_\text{v}$ is used to capture the relationships between different positions of the feature map and can be implemented using a 1×1 convolution.

During the training process, parameters are optimized by minimizing the loss function, and the threshold update obeys the same principle. However, it is crucial to choose an appropriate initial threshold. If the threshold at a specific position exceeds the absolute value of the feature at that position, the feature will be directly set to zero, thereby affecting further optimization. To obtain an appropriate initial threshold, the authors in [8] multiply normalize $\mathbf{Z}^a$ with the mean value of each position on the feature map, which can be expressed as follows:

$$\mathbf{Z}^{a'} = \sigma\left(\sum_{j=1}^{n} \rho(\mathbf{W}_\text{k}\mathbf{x}_j)\mathbf{W}_\text{v}\mathbf{x}_j^a\right) \cdot \frac{\sum_{j=1}^{n} \mathbf{x}_j^a}{n} \tag{4}$$

where $\sigma(\cdot)$ represents the normalization function, $\mathbf{Z}^{a'} \in \mathrm{R}^{h\times w\times 1}$, the algorithm proposed for threshold denoising is shown in Algorithm 1.

The above process calculates the thresholds for different positions in the feature map, obtaining a set of threshold maps with the same width and height as the input feature map. Subsequently, the computed threshold maps are employed to perform threshold computation on each feature channel. Due to the distinct importance of various feature channels in classification tasks [14], utilizing a single threshold map for all feature channels is inappropriate. Therefore, this paper considers applying global average pooling to each feature map, utilizing fully connected layers to capture cross-channel correlations, finally, employing the sigmoid function for normalization. The process can be summarized as follows:.

$$\mathbf{p} = \sigma\left(\text{FC}\left(\frac{\sum_{i=1}^{M} \mathbf{x}^i}{M}\right)\right) \tag{5}$$

where M represents the number of positions on the feature map, FC represents the fully connected operation for capturing inter-channel interaction relationships, $\mathbf{p} \in \mathrm{R}^{1\times 1\times n}$. The threshold at a specific position of the feature map is defined as:

$$\mathbf{Z}^{a''} = \sigma\left(\sum_{j=1}^{n} \rho(\mathbf{W}_\text{k}\mathbf{x}_j)\mathbf{W}_\text{v}\mathbf{x}_j^a\right) \cdot \frac{\sum_{j=1}^{n} \mathbf{x}_j^a}{n} \cdot \mathbf{p}. \tag{6}$$

As the increasing of network depth, noise features presented in data have been activated with network propagation and transmitted to deeper network layers. It would increase model parameters as well as computational complexity and even have detrimental effects on the recognition task. To address these issues, this paper introduces a denoising module in the shallow layer of the neural network. Therefore, the threshold at feature map position a can be defined as:

$$\mathbf{Z}_\text{low}^{a''} = \sigma\left(\sum_{j=1}^{n} \rho(\mathbf{W}_\text{k}\mathbf{x}_j)\mathbf{W}_\text{v}\mathbf{x}_j^a\right)_\text{low} \cdot \frac{\sum_{j=1}^{n} \mathbf{x}_j^a}{n} \cdot \mathbf{p}_\text{low} \tag{7}$$

where low represents shallow denoising.

3. Experimental Results

3.1 Setup

This paper conducts simulation experiments based on 12 types of jamming patterns, as shown in Fig. 2. The power of the jamming signals is set to 0 dBm. The baseband signals are shaped using a root-raised cosine filter with a roll-off factor 0.5. An 8PSK modulation scheme is employed, with a sampling frequency of 2000 MHz and a range of 600–920 MHz. Each jamming pattern generates 200 data sets, resulting in total of 2400 data sets. The training set comprises 70% of the data, while the remaining 30% is used for testing.

All the network models were implemented on a computer with Windows 10 Professional (64-bit) operating system, using an NVIDIA RTX 3080 GPU. The implementation was done using Tensorflow 2.6.0 and Keras 2.6.0 frameworks. The Adam optimizer was employed with an initial learning rate of 0.001, and the learning rate was reduced by 10% every 30 epochs. The training process involved a total of 150 iterations. The model structure and parameters are shown in Tab. 1.

Network	Layer	Activation	Channel num	Parameters
Convolutional block 1	Con2v	Relu	4	Kernel:5
	BN	-	-	-
	Con2v	Relu	4	Kernel:5
	Maxpooling	-	-	Stride:2
Threshold denoising block	Con1v	Softmax	1	Kernel:5
	Con1v	Sigmoid	1	Kernel:1
	Fc	-	-	-
Convolutional block 2	Con2v	Relu	8	Kernel:5
	BN	-	-	-
	Con2v	Relu	8	Kernel:5
	Maxpooling	-	-	Stride:2
Convolutional block 3	Con2v	Relu	16	Kernel:5
	BN	-	-	-
	Con2v	Relu	16	Kernel:5
	Maxpooling	-	-	Stride:2

Tab. 1. Model structure and parameters.

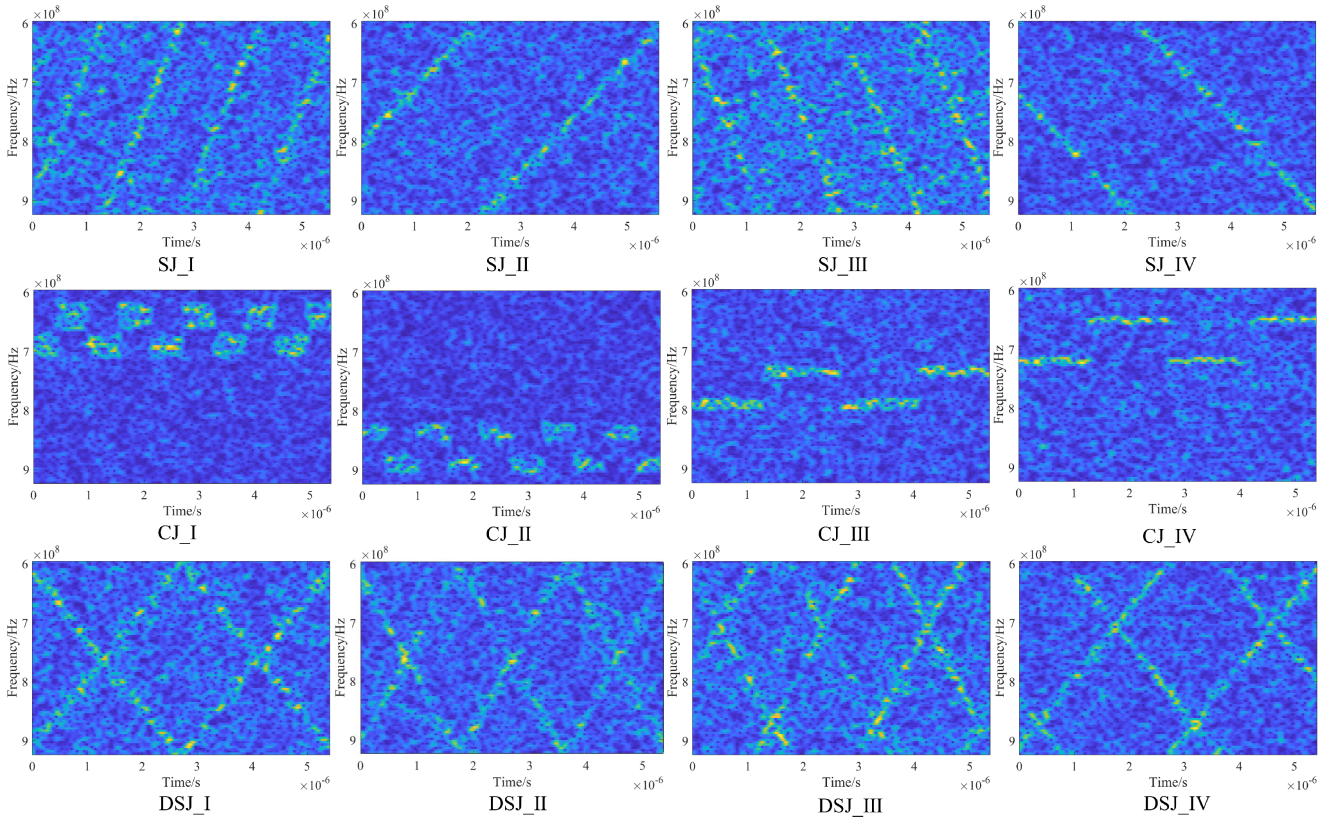


Fig. 2. Spectral waterfall plots of 12 jamming modes.

3.2 Performance Analysis

When recognizing jamming patterns, there is no denying that the noise environment presents a realistic and intricate challenge. The noise environment adds complexity to the model's input data and can obscure the recognition of jamming patterns, making it even more challenging. However, complexity issues should be considered when employing deep learning methods for addressing jamming pattern recognition problems. DNNs typically require a significant amount of computational resources during training. Additionally, as the number of model parameters increases, the computational complexity escalates, potentially limiting the feasibility in resource-constrained environments. Consequently, a balance must be struck between the number of parameters, computational complexity and performance to ensure that the model can achieve both effectiveness and accuracy in practical applications within noisy environments. Therefore, this paper investigates the impact of placing varying numbers of denoising modules at different network layers on recognition performance. Furthermore, it evaluates the number of parameters setting different quantities of denoising modules at distinct network layers and comprehensively considers the corresponding computational complexity.

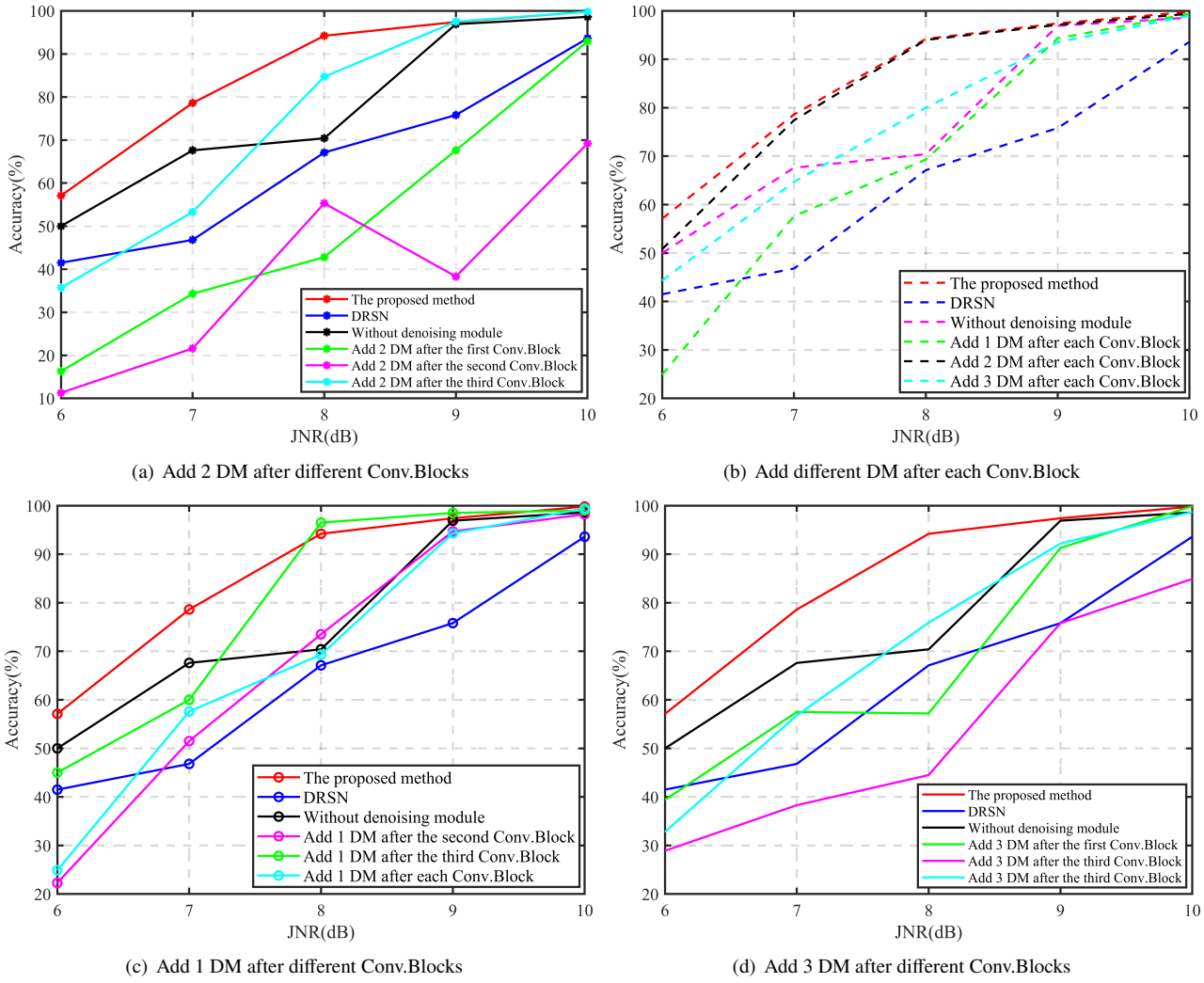
Table 2 demonstrates the computational costs of various models in the experiments. When introducing different numbers of DM after the same Conv.Block, as the DM increases, the training parameters and FLOPs also increase. Furthermore, when an equal number of DM is introduced cross the first, second and the third Conv.Block, the training parameters and FLOPs show a downward trend, the algorithm proposed in this paper exhibits a slightly higher parameter counts compared to add 1 DM after the second and the third Conv.Block.

Model	Trainable parameters	FLOPs
Without denoising module	0.593M	92.561M
The proposed method	0.649M	93.002M
Add 1 DM after the second Conv.Block	0.621M	92.994M
Add 1 DM after the third Conv.Block	0.607M	92.990M
Add 1 DM after each Conv.Block	0.691M	93.864M
Add 2 DM after the first Conv.Block	0.705M	93.443M
Add 2 DM after the second Conv.Block	0.649M	93.426M
Add 2 DM after the third Conv.Block	0.621M	93.419M
Add 2 DM after each Conv.Block	0.788M	95.166M
Add 3 DM after the first Conv.Block	0.760M	93.884M
Add 3 DM after the second Conv.Block	0.677M	93.859M
Add 3 DM after the third Conv.Block	0.635M	93.848M
Add 3 DM after each Conv.Block	0.886M	96.468M

*DM denote Denoising Module

Tab. 2. The computational costs of various models in the experiments.

Figure 3 displays the recognition rates achieved by adding different DM after various Conv.Blocks and the advanced algorithm DRSN [9]. In Fig. 3(a) and (d), the recognition rates are compared when two and three DM are added to each Conv.Block, respectively. It can be observed from the figures that the proposed model is superior to other models.



(a) Add 2 DM after different Conv.Blocks

(b) Add different DM after each Conv.Block

(c) Add 1 DM after different Conv.Blocks

(d) Add 3 DM after different Conv.Blocks

Fig. 3. The recognition accuracy with different DM added after different Conv.Blocks.

In Fig. 3(b), the recognition rates are compared when 1, 2, and 3 DM are added after each Conv.Block, along with the proposed model and a scenario without DM and DRSN. It is evident from the graph that the proposed model achieves the highest recognition rate. As shown in Fig. 3(c), at 8 dB and 9 dB, the recognition rate is slightly higher when adding one DM after the third Conv.Block than the proposed model. However, both models' recognition rates differ by 2.3% and 0.9% at these JNRs. Nevertheless, at other JNRs, the performance of the proposed model surpasses the others. The shallow DM proposed in this paper balances parameter counts, computational complexity, and recognition performance, ensuring that it can effectively address the challenges posed by noisy environments without excessively taxing computational resources.

Figure 4 presents the confusion matrix of classification results at a JNR of 8 dB when using the shallow denoising approach. SJ_I, SJ_II, SJ_III, SJ_IV, CJ_I, CJ_II, CJ_III, CJ_IV, DSJ_I, DSJ_II, DSJ_III and DSJ_IV respectively represent sweep jamming I, sweep jamming II, sweep jamming III, sweep jamming IV, comb sweep jamming I, comb sweep jamming II, comb sweep jamming III, comb sweep jamming IV, double sweep jamming I, double sweep jamming II, double sweep jamming III, double sweep jamming IV. The colour bar indicates that the darker the colour, the higher the recognition rate. Results demonstrate that in addition to SJ_I, SJ_II, SJ_III, SJ_IV, and CJ_IV, all others can achieve a high recognition rate, that is because SJ can only observe information on one frequency point, while DSJ have multiple frequency points compared to SJ. Although CJ has a frequency point, CJ_I, CJ_II, and CJ_III have wide bandwidth. CJ_IV has the same bandwidth as SJ, causing CJ_IV to be mistakenly identified as SJ, resulting in a low recognition rate. Figure 5 shows the loss and accuracy of the training set at JNR = 8 dB.

In summary, the analysis of classification accuracy, computational complexity, and parameter quantity can demonstrate the effectiveness of shallow denoising methods.

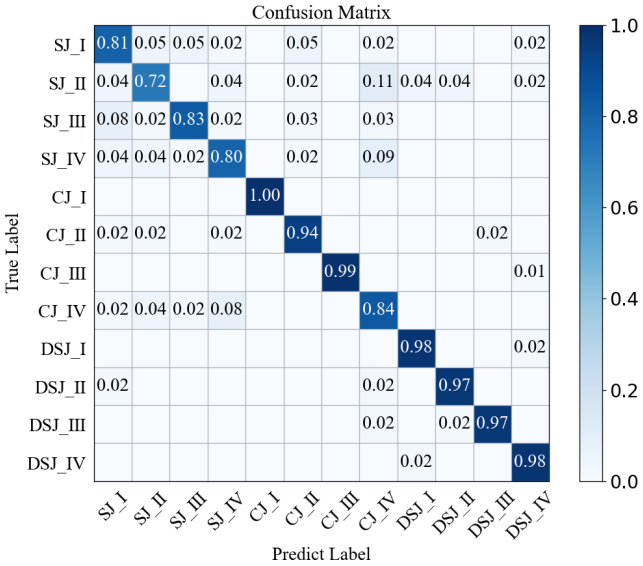
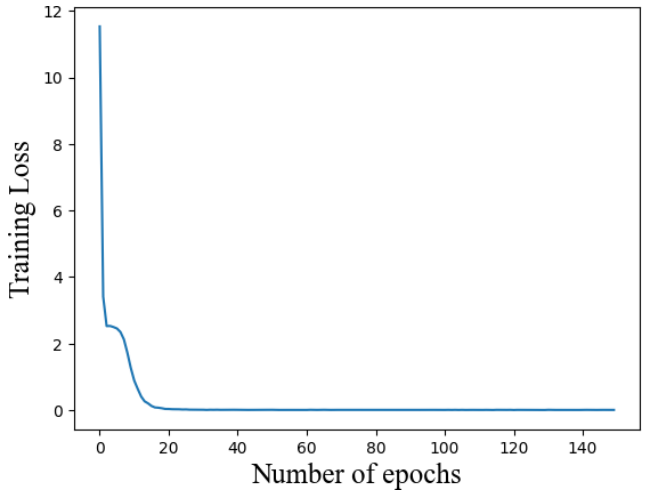
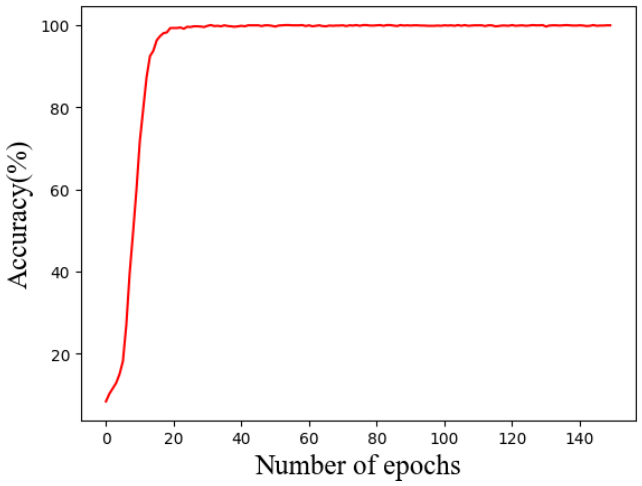


Fig. 4. The confusion matrix for shallow denoising at a JNR of 8 dB.



(a) The loss curve of the proposed method



(b) The accuracy curve of the proposed method

Fig. 5. Loss and accuracy results of the proposed method.

4. Conclusion

A practical jamming recognition algorithm is expected to perform well under noisy conditions. In response to this issue, this paper innovatively proposes a shallow denoising method based on deep learning. This method achieves automatic feature denoising by reducing the activation of noise in the feature space. A comprehensive experimental analysis further verified the effectiveness of the proposed method. It is worth noting that the proposed method performs better under low JNR conditions than the comparative method. The experimental results demonstrate that the proposed method has specific practical value. At the same time, machine learning-based noise processing methods are a promising research direction.

Acknowledgments

The work is supported by the National Natural Science Foundation of China (U22B2002, No. 61971439) and the National Key Laboratory of Wireless Communications Foundation under Grant IFN20230207.

References

[1] LIU, X., XU, Y. H., JIA, L. L., et al. Anti-jamming communications using spectrum waterfall: A deep reinforcement learning approach. *IEEE Communications Letters*, 2018, vol. 22, no. 5, p. 998–1001. DOI: 10.1109/lcomm.2018.2815018

[2] XIAO, L., LU, X. Z., XU, T. W., et al. Reinforcement learning-based mobile offloading for edge computing against jamming and interference. *IEEE Transactions on Communications*, 2020, vol. 68, no. 10, p. 6114–6126. DOI: 10.1109/tcomm.2020.3007742

[3] SHAO, G. Q., CHEN, Y. S., WEI, Y. S. Convolutional neural network-based radar jamming signal classification with sufficient and limited samples. *IEEE Access*, 2020, vol. 8, p. 80588–80598. DOI: 10.1109/access.2020.2990629

[4] WANG, Y. F., SUN, B., WANG, N. Recognition of radar active-jamming through convolutional neural networks. *The Journal of Engineering*, 2019, vol. 2019, no. 21, p. 7695–7697. DOI: 10.1049/joe.2019.0659

[5] QU, Q. Z., WEI, S. J., LIU, S., et al. JRNet: Jamming recognition networks for radar compound suppression jamming signals. *IEEE Transactions on Vehicular Technology*, 2020, vol. 69, no. 12, p. 15035–15045. DOI: 10.1109/tvt.2020.3032197

[6] LIU, S. Y., XU, Y. F., CHEN, X. Q., et al. Pattern-aware intelligent anti-jamming communication: A sequential deep reinforcement learning approach. *IEEE Access*, 2019, vol. 7, p. 169204–169216. DOI: 10.1109/access.2019.2954531

[7] HAN, H., LI, W., FENG, Z. B., et al. Proceed from known to unknown: Jamming pattern recognition under open-set setting. *IEEE Wireless Communications Letters*, 2022, vol. 11, no. 4, p. 693–697. DOI: 10.1109/lwc.2021.3140145

[8] CHEN, Y. F., ZHU, L., YAO, C. H., et al. Global context-based threshold strategy for drone identification under the low SNR condition. *IEEE Internet of Things Journal*, 2023, vol. 10, no. 2, p. 1332–1346. DOI: 10.1109/jiot.2022.3205065

[9] ZHAO, M. H., ZHONG, S. S., FU, X. Y., et al. Deep residual shrinkage networks for fault diagnosis. *IEEE Transactions on Industrial Informatics*, 2020, vol. 16, no. 7, p. 4681–4690. DOI: 10.1109/tii.2019.2943898

[10] XIE, C. H., WU, Y. X., VAN DER MAATEN, L., et al. Feature denoising for improving adversarial robustness. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach (CA, USA), 2019, p. 501–509. DOI: 10.1109/CVPR.2019.00059

[11] PENG, G. Z., CHEN, W. J. Fringe pattern inpainting based on dual-exposure fused fringe guiding CNN denoiser prior. *Optica Applicata*, 2022, vol. 52, no. 2, p. 179–193. DOI: 10.37190/oa220203

[12] ZHANG, X. Y., MANZI, M., VOGELS, T., et al. Deep compositional denoising for high-quality Monte Carlo rendering. *Computer Gaphics Forum*, 2021, vol. 40, no. 4, p. 1–13. DOI: 10.1111/cgf.14337

[13] DONG, W. S., WANG, P. Y., YIN, W. T., et al. Denoising prior driven deep neural network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019, vol. 41, no. 10, p. 2305–2318. DOI: 10.1109/TPAMI.2018.2873610

[14] HU, J., SHEN, L., ALBANIE, S., et al. Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020, vol. 42, no. 8, p. 2011–2023. DOI: 10.1109/TPAMI.2019.2913372

About the Authors ...

Changhua YAO received his B.S. degree in Automation from Zhejiang University in 2005 and Ph.D. degree in Communications and Information Systems from PLA University of Science and Technology in 2016. He was a post doctor at PLA University of Science and Technology in 2017–2019. He is now a professor at the School of Electronic and Information Engineering, Nanjing University of Information Science and Technology. His research interests focus on intelligent unmanned swarm, smart wireless networks, electromagnetic spectrum antagonism.

Yang LI received the B.S. degree from Shangqiu Normal University in 2020, Shangqiu, China, where she is currently pursuing a master's degree with the School of Electronic and Information Engineering, Nanjing University of Information Science and Technology. Her research interests include deep learning and pattern recognition.

Yufan CHEN (corresponding author) received the B.S. degree from the Army Engineering University of PLA, Nanjing, China, in 2018, where he is currently pursuing the Ph.D. degree with the Institute of Communications Engineering. His current research interests include OOD detection, open-set recognition and drone identification.

Kaixin CHENG received the Ph.D. degree in Cyberspace Security from the College of Communications Engineering, Army Engineering University of PLA, Nanjing, China in 2022. Her research interests include communication network security, spectrum behavior sensing and artificial intelligence.

Appendix A: Algorithm 1

Algorithm 1. Threshold denoising algorithm.

Input:
Input feature map: $\mathbf{x}$;
Number of feature channels: n;
Leachable parameters: $\mathbf{W}_\mathrm{k}, \mathbf{W}_\mathrm{v}$;
Output:
Threshold at position a: $\mathbf{Z}^a$;
$\mathrm{sum}_2 \leftarrow 0$
for $j = 1 \rightarrow n$ **do**
  $\mathrm{sum}_1 \leftarrow 0$
  for $m = 1 \rightarrow n$ **do**
    $\mathrm{sum}_1 \leftarrow \mathrm{sum}_1 + \exp(\mathbf{W}_\mathrm{k}\mathbf{x}_m)$
  end for
  $\mathrm{sum}_2 = \mathrm{sum}_2 + \frac{\exp(\mathbf{W}_\mathrm{k}\mathbf{x}_j)}{\mathrm{sum}_2} \cdot \mathbf{W}_\mathrm{v}\mathbf{x}_j$
end for
Feature value at position a of feature map j: x_j^a;
Feature value at position i of the feature map: x^i;
$\mathrm{avg}_1 \leftarrow 0$
$\mathrm{avg}_2 \leftarrow 0$
for $j = 1 \rightarrow n$ **do**
  $\mathrm{avg}_1 \leftarrow \mathrm{avg}_1 + x_j^a$
end for
for $j = 1 \rightarrow M$ **do**
  $\mathrm{avg}_2 \leftarrow \mathrm{avg}_2 + x^i$
end for
$\mathbf{Z}^a \leftarrow \mathrm{Sigmoid}(\mathrm{sum}_2) * \frac{\mathrm{avg}_1}{n} * \mathrm{Sigmoid}(\frac{\mathrm{avg}_2}{M})$