Received: 17 December 2017 | Revised: 5 March 2019 | Accepted: 7 April 2019

DOI: 10.4218/etrij.2017-0327

ORIGINAL ARTICLE

WILEY ETRI Journal

Ensemble convolutional neural networks for automatic fusion recognition of multi-platform radar emitters

Zhiwen Zhou[1] | **Gaoming Huang**[2] | **Xuebao Wang**[2]

[1]Department of Operational Command, Command College of the PAP, Tianjin, China

[2]College of Electronic Engineering, Naval University of Engineering, Wuhan, China

Correspondence
Zhiwen Zhou, Department of Operational Command, Command College of the PAP, Tianjin, China.
Email: mini_paper@sina.com

Funding information
This research was supported by the National High Technology 863 Program (No. 2014AA7014061) and the National Natural Science Foundation of China (No. 61501484).

Abstract

Presently, the extraction of hand-crafted features is still the dominant method in radar emitter recognition. To solve the complicated problems of selection and updation of empirical features, we present a novel automatic feature extraction structure based on deep learning. In particular, a convolutional neural network (CNN) is adopted to extract high-level abstract representations from the time-frequency images of emitter signals. Thus, the redundant process of designing discriminative features can be avoided. Furthermore, to address the performance degradation of a single platform, we propose the construction of an ensemble learning-based architecture for multi-platform fusion recognition. Experimental results indicate that the proposed algorithms are feasible and effective, and they outperform other typical feature extraction and fusion recognition methods in terms of accuracy. Moreover, the proposed structure could be extended to other prevalent ensemble learning alternatives.

KEYWORDS

deep learning, emitter recognition, ensemble learning, robustness, time-frequency analysis

1 | INTRODUCTION

In modern warfare, emitter recognition is one of the primary functions of electronic intelligence (ELINT) and ESM systems. It can provide timely information on emitters and aid decision making. In hostile situations, the category, usage, and state information of intercepted signals from radars are comprehensively analyzed. Then, the results are reported to operational personnel [1]. The deinterleaving process generally exploits the pulse descriptor word (PWD), and precise measurements are used for the subsequent classification. However, with the increasing complexity and density of electromagnetic signals, conventional recognition techniques no longer meet practical demands. Moreover, although the existing approaches (eg, template matching) have been progressively improved, the imprecise measurements, development of new system radars, and low signal-to-noise ratios (SNRs) still affect the recognition performance. Meanwhile, a single surveillance platform or system typically suffers from information deficiency and false alarms, which definitely affects combat capabilities. Therefore, it is challenging to extract effective features in order to achieve high accuracy, especially with low SNRs and in a complex environment.

Presently, feature-based (FB) algorithms prevail in the field of emitter recognition. For the feature extraction stage, various features were proposed to achieve high accuracies. Guo et al [2] proposed the extraction of ambiguity function main ridge slice features and denoising with singular value decomposition(SVD), which was preferable for low SNRs. Zeng et al [3] applied ZAM-GTFR to form generalized time-frequency representations and achieved more than 90% accuracy when the SNR was lower than –2 dB. Ma et al [4] used random projection to compress signals and obtained good robustness for low

SNRs. However, it cannot be denied that although the extracted features could identify certain types of emitters, its applicability to other types and the efficiency of experience-based methods remain to be studied. In addition, it is difficult to compare features, and the design process is time-consuming. However, only improving the performance of a single system is far from practical requirements once it severely degrades. Multi-sensor fusion [5] of emitter signals can be implemented by combining supplementary and contradictory information from multiple systems. Multi-collaborative representations [6] and Dezert-Smarandache theory [7] were proposed to address the problem of fusion recognition.

In recent years, deep learning (DL) has been widely applied in several fields owing to its ability to generate intermediate representations in a hierarchical manner. Compared with existing shallow methods, DL extracts high-level, abstract, and inherent features to describe an original data distribution [8]. Typical DL models include stacked autoencoders (SAEs), deep Boltzmann machines (DBMs), and CNNs [9]. To the best of our knowledge, DL has been studied for in emitter waveform recognition a little, and its application is simple. References [10–12] directly adopted a CNN as the feature extractor to achieve recognition. However, the CNN was not fully exploited and the scenario of multiple emitters was not considered. For multi-platform fusion, ensemble learning [13] offers reduced error rates and a fine generalization property. The assumption of ensemble learning is that each classifier is conditionally independent, which is common for a multi-sensor system, especially for heterogeneous sensors. Inspired by the aforementioned studies, we construct a DL-based architecture. The architecture uses a 2D-oriented CNN to extract deep features and produce fusion inference. Ensemble methods, namely boosting and bagging [14], are introduced to prompt fusion recognition; they are denoted as CNNBoosting and CNNBagging, respectively. Our main contributions are as follows:

- We propose to exploit deep features of radar emitters automatically rather than empirical features. As CNNs are suitable for 2D entries, the time-frequency distribution of radar emitters provides an alternative.
- We integrate CNNs into an ensemble learning framework to construct a multi-platform fusion architecture. The ensemble CNNs utilize the decision-level fusion via the trained weights.

The remainder of the paper is organized as follows. Section 2 introduces the fundamentals of emitter time-frequency representations. Sections 3 and 4 detail the multi-platform fusion recognition algorithms. Simulation experiments and analyses are given in Section 5, and Section 6 provides the conclusions.

2 | MODEL AND REPRESENTATION OF EMITTER SIGNALS

The mathematical model of intercepted emitter signals in the time domain can be described as follows,

$$s(t) = A(t)\exp\left[j2\pi f_0 t + jc(t)\right] + n(t), \tag{1}$$

where f_0 denotes the carrier frequency, and $A(t)$ and $c(t)$ are the amplitude and phase function, respectively. $A(t)$ is simplified as a constant for slow hopping compared with f_0 and $n(t)$. Because the time-frequency representation for each intentional modulation is unique, and because the emitter in the time-frequency domain provides more feature information, signals are processed in the time-frequency domain. Hence, the definition of the transformed signal is given with a short-time Fourier transformation (STFT),

$$\mathrm{STFT}_s(n,\omega) = \sum_{m=-\infty}^{+\infty} s(m)g(n-m)e^{-j\omega m}, \tag{2}$$

where $g(m)$ is a predefined window function, and $s(m)$ is the discrete-time signal. Then, equal frequency interval sampling is implemented for (2), $\omega_r = 2\pi r/N$ ($r = 0, 1, \ldots, N-1$), which yields,

$$\mathrm{STFT}_s(n,r) = \sum_{m=-\infty}^{+\infty} s(m)g(n-m)e^{-j\frac{2\pi r}{N}m}. \tag{3}$$

Substituting $(l + n)$ for m, which produces,

$$\begin{aligned}\mathrm{STFT}_s(n,r) &= e^{-j\frac{2\pi r}{N}n}\sum_{l=-\infty}^{+\infty} e^{j\omega_0(l+n)+jc(l+n)}g(-l)e^{-j\frac{2\pi r}{N}l} \\ &= e^{j(\omega_0-\omega_r)n}\sum_{l=-\infty}^{+\infty} e^{jc(l+n)}g(-l)e^{j(\omega_0-\omega_r)l}.\end{aligned} \tag{4}$$

From (4), it is clear that $\mathrm{STFT}_s(n, r)$ fundamentally depends on $c(m)$ for a given $g(m)$, which implies that time-frequency distributions of different modulations are determined by $c(t)$. Thayaparan et al [15] pointed out that the time-frequency imaging of radar signals provides a solution to unique classification. Therefore, we adopt 2D time-frequency imaging representations for the subsequent processing.

3 | ENSEMBLE CONVOLUTIONAL NEURAL NETWORKS FOR EMITTER RECOGNITION

Most state-of-the-art methods for emitter recognition focus on the artificial extraction of intra-pulse features or mono-platform recognition. Instead, this section emphasizes deep features and decision-level fusion recognition. For each independent platform, the corresponding CNN module is pre-trained and fine-tuned with training samples to learn network parameters. The ensemble weights are determined by multiple CNNs having different properties, which are used to

collaboratively infer the labels of testing signals in the fusion layer.

3.1 | Algorithm architecture

As illustrated in Figure 1, the proposed architecture comprises three parts, namely signal pre-processing, multi-branch CNN modules, and decision-level fusion recognition. To describe the pre-processing stage briefly, T original signals that belong to an identical type but different sequence pulses are randomly input into independent branches. Then, T-branch signals are further processed in the time-frequency domain, denoted as $P_1, \ldots, P_T$.

To strengthen the correlations and avoid high-dimensional computations, the 2D time-frequency images $\{\mathbf{S}^{(t)} \in \mathbb{R}^{N_1 \times N_2}, t = 1, 2, \ldots, T\}$ are segmented into small 2D patches in the same manner, where $\mathbf{S}^{(t)}$ denotes the time-frequency representation. The contrast normalization is followed by mapping the input patches into a canonical form, which reduces data redundancy and accelerates the convergence speed of the single CNN,

$$\mathbf{S}^{(t)} = \frac{\mathbf{S}^{(t)} - \mathbb{E}\left[\mathbf{S}^{(t)}\right]}{\mathbb{D}\left[\mathbf{S}^{(t)}\right]} \tag{5}$$

where $\mathbb{E}(\cdot)$ and $\mathbb{D}(\cdot)$ denote the mean and variance operations, respectively. Because conventional algorithms train on non-whitened data, and because the whitening processing is beneficial for learning discriminative representations, we adopt ZCA whitening [16] to process the normalized data. It yields

$$\mathbf{S}_{\mathbf{W}}^{(t)} = \mathbf{W}\mathbf{S}^{(t)} \tag{6}$$

where $\mathbf{W}$ is the whitening matrix. Normally, $\mathbf{W} = \mathbf{U}\mathbf{V}^{-1/2}\mathbf{U}^{\mathrm{T}}$, $\mathbf{U}$ and $\mathbf{V}$ denote eigenvectors and eigenvalues of the covariance of $\mathbf{S}^{(t)}$, respectively. All the processed samples are stacked into a 3D training set $\mathbf{A} \in \mathbb{R}^{N_1 \times N_2 \times Q}$, where Q is the sample number. The preprocessed 3D data are fed into each CNN to train the hierarchical structure in an unsupervised manner. Then, weights are generated according to the properties of the CNNs with ensemble methods. For the proposed ensemble CNNs, they are operated in a supervised manner as the training labels are indispensable.

As shown in Figure 1, T CNNs are of the same structure, but with different network parameters. Generally, a CNN comprises a convolutional layer, an activation function, a pooling layer, and a fully-connected layer. To elaborate, the preprocessed input ($\mathbf{S}_{\mathbf{W}}^{(t)} \in \mathbb{R}^{64\times64}$) is convoluted with randomly initialized kernels $\mathbf{F} \in \mathbb{R}^{5\times5}$, and a bias term $\boldsymbol{b}$ is added to the resultant feature maps. Instead of a sigmoid function in general, the ReLU activation function $g(x) = \max(0, x)$ is a more advisable alternative to boost the nonlinear pointwise network. It has been proven to speed up the convergence significantly and increase the descriptive ability of the network [17]. Finally, the max-pooling layer is utilized to reduce the sensitivity to small input shifts.

Then, given T CNNs, the kth layer output state of the tth branch can be formulated as follows:

$$\mathbf{L}_k^t = \mathrm{pool}\left(\mathrm{ReLU}\left(\mathbf{L}_{k-1}^t \otimes \mathbf{F}_k^t + \boldsymbol{b}_k^t\right)\right) \tag{7}$$

where $\otimes$ is the 2D convolutional operation, and $\mathrm{pool}(\cdot)$ denotes the pooling operation. The network parameters, that is, $\boldsymbol{\theta} = \left\{\mathbf{F}_k^t, b_k^t, t = 1, \ldots, T, k = 1, \ldots, K\right\}$ are iteratively optimized by minimizing the classification error over the training set. Commonly, the stochastic gradient descent (SGD) algorithm is adopted to achieve this. In SGD, the network parameters are learned by the derivatives of $\mathbf{F}$ and $\boldsymbol{b}$. The learning performance is related to the slight learning rate ω during optimization. Once the single CNN parameters are well trained and the multiple CNNs are weighted with ensemble methods, the decision-level fusion recognition can be executed for given multiple input vectors $\{\mathbf{x}_t, t = 1, \ldots, T\}$, which yields,

$$\begin{aligned} \mathrm{H}(\mathbf{Y}_{\mathrm{fusion}}) &= \arg\max_{i\in\{1,\ldots,C\}} \left[\sum_{t=1}^{T} \beta_t \Pr\left(Y_t = i|\boldsymbol{\theta}_t\right)\right] \\ &= \arg\max_{i\in\{1,\ldots,C\}} \left[\sum_{t=1}^{T} \beta_t \frac{\exp\left(\mathbf{W}^t\mathbf{x}_t + \boldsymbol{b}^t\right)}{\sum_i^C \exp\left(\mathbf{W}_i^t\mathbf{x}_t + \boldsymbol{b}_i^t\right)}\right], \end{aligned} \tag{8}$$

where $\{\beta_t\}$ are produced via ensemble methods, $\boldsymbol{\theta}_t = \{\mathbf{W}_t, \boldsymbol{b}_t, t = 1, \ldots, T\}$ is the network parameter set of the decision-making logistic regression layer, $\mathbf{W}_t$ and $\boldsymbol{b}_t$ are the weights and biases, and C is the total number. The label information can be therefore deduced by the weighted probability of each class. In this study, we propose two algorithms aiming to achieve multi-platform fusion recognition of radar emitters.

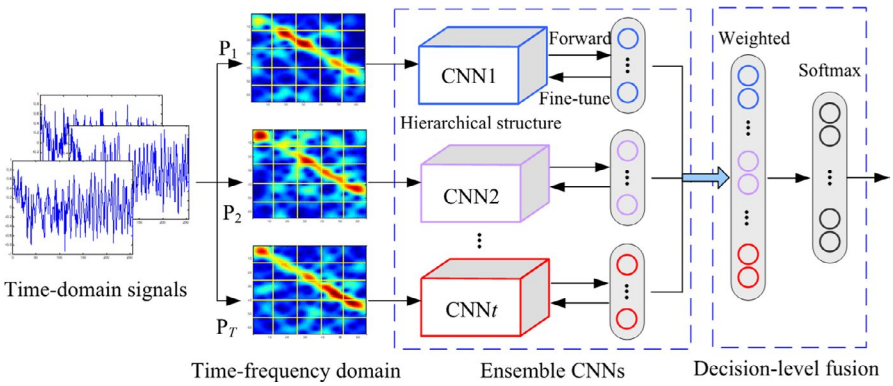

FIGURE 1 Architecture of ensemble CNNs for emitter recognition

3.2 | Boosting-based CNNs (CNNBoosting)

The processing branches are composed of T CNNs, but the intercepted signals and CNNs structures are not identical. The difference between them is measured by weights with boosting in this subsection. Unlike conventional boosting algorithms, such as AdaBoost, which aim to control the selection of multiple weak classifiers to form a strong one [18], we expect to fuse multiple signals to produce a more convincing recognition. As the resource data are obtained from identical types of signals, boosting is performed to train multiple CNNs and assign weights in the reference stage. However, the key to the algorithm scheme is to minimize the training error with a pre-assigned number of iterations. Similar to AdaBoost.M2, which is a variant of AdaBoost, we propose a multi-class fusion recognition algorithm, CNNBoosting. In the training stage, multiple CNNs are trained as weak learners and fine-tuned to minimize the weighted error. Meanwhile, optimum weights are selected. Subsequently, we combine the predictions of multiple CNNs to achieve a boosting-based prediction in the reference stage. The proposed CNNBoosting is described in detail in Algorithm 1. Notably, the final predictive inference is determined via a majority voting strategy.

Algorithm 1 CNNBoosting Algorithm

Input: Labelled training set $\mathbf{A}\in\mathbb{R}^{N_1\times N_2\times Q}$, labels $\mathbf{Y}_{\text{tr}}$, iteration number T, CNN optimization parameters $\boldsymbol{\omega}$, label number C;

Training:

Initialize the distribution uniformly: $D_t^{(q)}=1/Q$.

Loop: **For** $t=1\to T$

Step 1: Train CNN t with $\mathbf{A}$ and $\mathbf{Y}_{\text{tr}}$ while randomly selecting ω_t;

Step 2: Predict the labels of validation set by the maximum probability of output perceptrons $\mathbf{Y}_{\text{pre},q}=\arg\max_{i=1,\ldots,C}\left\{\Pr\left(\mathbf{A}_q=i\right)\right\}, q=1,\ldots,Q$;

Step 3: Calculate the error on $D_t^{(q)}$, $\varepsilon_t=b_t D_t^{(q)}$, where $b_t=\max_{i\neq c}\Pr\left(\mathbf{Y}_{\text{pre},q}=i\right)-\Pr\left(\mathbf{Y}_{\text{pre},q}=c\right)$ and c is the actual label;

Step 4: Choose $\beta_t\in\mathbb{R}$ as: $\beta_t=\frac{1}{2}\log\left[(C-1)\frac{1-\varepsilon_t}{\varepsilon_t}\right]$;

Step 5: Update distribution $D_{t+1}^{(q)}=D_t^{(q)}\exp\left(\beta_t b_t\right)/Z_t$, where Z_t is the normalization factor to make $\sum_q^Q D_{t+1}^{(q)}=1$;

End

Output: $\{\beta_t\}_{t=1}^T$; Then the recognition is subsequently executed with weighted probability and majority voting.

3.3 | Bagging-based CNNs (CNNBagging)

Similar to CNNBoosting, an analogous algorithm called CNNBagging is proposed, which differs in terms of training strategy. The core of bagging lies in the bootstrap replications of the training set in a pseudo manner. It produces multiple base classifiers with diversity [19]. However, the final decision depends on the preset bootstrap samples. Although the bagging strategy seems somewhat simplistic, its property of reducing the entire estimate variance is suitable for recognition in this work. Furthermore, it is a suitable choice to combine multiple CNNs that are complex and unstable in a bagging-based manner compared with other weak learners. During the training process, CNNs are trained by randomly generating indices to cross-validate the training set. Simultaneously, the weights of multiple platforms are computed according to the classification error. Consequently, the decision-level prediction is inferred via majority voting, as described in Algorithm 2. It is evident that both the proposed algorithms are of the same structure, in which training and weights selection are offline while inference is on-line.

Algorithm 2 CNNBagging Algorithm

Input: Labeled training set $\mathbf{A}\in\mathbb{R}^{N_1\times N_2\times Q}$, labels $\mathbf{Y}_{\text{tr}}$, iteration number T, CNN optimization parameters $\boldsymbol{\omega}$, bootstrap ratio η;

Training:

Loop: **For** $t=1\to T$

Step 1: Randomly select $\eta\times Q$ samples out of $\mathbf{A}$ and $\mathbf{Y}_{\text{tr}}$, denoted as $\mathbf{A}_{\text{br}}$ and $\mathbf{Y}_{\text{br}}$;

Step 2: Train CNN t using $\mathbf{A}_{\text{br}}$ and $\mathbf{Y}_{\text{br}}$;

Step 3: Calculate the right predictions $\mathrm{R}=\left[\sum\arg\max_{i=1,\ldots,C}\left\{\Pr\left(\mathbf{A}_{\text{br},q}=i\right)\right\}=\mathbf{Y}_{\text{br},q}\right]$;

Step 4: Calculate right ratio within the whole samples $r=\frac{2\mathrm{R}}{Q(1-\eta)}-1$;

Step 5: Choose $\beta_t\in\mathbb{R}$ as: $\beta_t=\frac{1}{2}\log\frac{1+r}{1-r}$;

End

Output: $\{\beta_t\}_{t=1}^T$; The same method is adopted to infer the labels of testing samples according to the maximum weighted probability.

3.4 | Discussions

In this subsection, we investigate the proposed algorithms by analyzing the classification error on the training set. As the weight-updating mechanism is the kernel of both algorithms, we can expand the description of the training process. To clearly elaborate on the performance, the two algorithms are discussed separately. To gauge the learning performance of CNNBoosting, we first compute the probability of error classification

$$\begin{aligned}&\Pr_D\left[\mathrm{H}(\mathbf{Y}_{\text{fusion}})\neq c\right]\\&=\frac{1}{Q}\sum_{q=1}^{Q}\left\{c\neq\arg\max_i\left[\sum_{t=1}^{T}\beta_t\Pr\left(\mathbf{Y}_{\text{pre},q}=i\right)\right]\right\}\\&=\frac{1}{Q}\sum_{q=1}^{Q}\left\{\sum_{t=1}^{T}\beta_t\Pr\left(\mathbf{Y}_{\text{pre},q}=c\right)\leq\max_{i\neq c}\left[\sum_{t=1}^{T}\beta_t\Pr\left(\mathbf{Y}_{\text{pre},q}=i\right)\right]\right\},\end{aligned}\tag{9}$$

$$\leq\frac{1}{Q}\sum_{q=1}^{Q}\left\{\sum_{t=1}^{T}\beta_t\Pr\left(\mathbf{Y}_{\text{pre},q}=c\right)\leq\sum_{t=1}^{T}\beta_t\max_{i\neq c}\Pr\left(\mathbf{Y}_{\text{pre},q}=i\right)\right\},\tag{10}$$

$$\leq\frac{1}{Q}\sum_{q=1}^{Q}\exp\left\{\sum_{t=1}^{T}\beta_t\left[\max_{i\neq c}\Pr\left(\mathbf{Y}_{\text{pre},q}=i\right)-\Pr\left(\mathbf{Y}_{\text{pre},q}=c\right)\right]\right\}.\tag{11}$$

For any m and n, the inequality $\sum(m-n)\leq\exp\sum(m-n)$ holds. Then, a similar invariant is consequently obtained for (11). According to Step 5 in Algorithm 1, we unwrap the distribution D_T as

$$\begin{aligned}D_T&=\prod_{t=1}^{T}\left(\frac{e^{\beta_{T-1}b_{T-1}}}{Z_{t-1}}\cdots\frac{e^{\beta_1 b_1}}{Z_1}\right)\\&=\frac{D_1}{\prod_{t=1}^{T}Z_t}\exp\left(\sum_{t=1}^{T}\beta_t b_t\right).\end{aligned}\tag{12}$$

Then, substituting $D_1 = 1/Q$ and (12) into (11), it yields,

$$\Pr_D\left[H\left(\mathbf{Y}_{\text{fusion}}\right)\neq c\right]\leq\sum_{q=1}^{Q}D_T\times\prod_{t=1}^{T}Z_t=\prod_{t=1}^{T}Z_t.\tag{13}$$

Then, (13) implies that we can design a reasonable approach to greedily approximate the bound by minimizing Z_t. In each round of CNNBoosting, β_t and weak learners are deliberately learned and updated. The mathematic upper bound of $\Pr_D[\mathrm{H}(\mathbf{Y}_{\text{fusion}}) \neq c]$ serves as a reference to reach, and it is imprecise and difficult to design in practice. Then, although the product of normalization factors provides a theoretical value, the experimental results in Section 5 were satisfactory compared with those of other algorithms.

To investigate how CNNBagging works, we deduce a similar proof to Tao [20] as follows. Assume that each $(Y, \boldsymbol{x})$ is the sample data from the learning set $\mathbf{A}$, where Y denotes the numerical labels of $\boldsymbol{x}$. Furthermore, samples are independently drawn from a probability distribution P. Suppose $\sigma(\boldsymbol{x}, \mathbf{A})$ is the predictor of each CNN branch; then the aggregated predictor can be given according to the average one over $\mathbf{A}$,

$$\sigma_{\mathbf{A}}(\boldsymbol{x},P)=E_{\mathbf{A}}\sigma(\boldsymbol{x},\mathbf{A}).\tag{14}$$

As the average predictor error can be derived, $e_{\text{ave}} = \mathrm{E}_{\mathbf{A}}\mathrm{E}_{Y,\boldsymbol{x}}(Y - \sigma(\boldsymbol{x}, \mathbf{A}))^2$. Then, the aggregated predictor error is estimated by

$$e_{\text{agg}}=\mathrm{E}_{Y,\boldsymbol{x}}(Y-\sigma(\boldsymbol{x},P))^2.\tag{15}$$

Using the inequality $\frac{1}{M}\sum_{m=1}^{M}(z_m)^2\geq\left(\frac{1}{M}\sum_{m=1}^{M}z_m\right)^2$ and unwrapping (15), we know

$$\begin{aligned}e_{\text{ave}}&=\mathrm{E}_{Y,\boldsymbol{x}}Y^2-2\mathrm{E}_{Y,\boldsymbol{x}}\sigma_{\mathbf{A}}+\mathrm{E}_{Y,\boldsymbol{x}}\mathrm{E}_{\mathbf{A}}\sigma^2(\boldsymbol{x},\mathbf{A})\\&\geq\mathrm{E}_{Y,\boldsymbol{x}}y^2-2\mathrm{E}_{Y,\boldsymbol{x}}\sigma_{\mathbf{A}}+\mathrm{E}_{Y,\boldsymbol{x}}\left[\mathrm{E}_{\mathbf{A}}\sigma(\boldsymbol{x},\mathbf{A})\right]^2\\&=\mathrm{E}_{Y,\boldsymbol{x}}\left(y-\sigma_{\mathbf{A}}\right)^2=e_{\text{agg}}.\end{aligned}\tag{16}$$

Hence, (16) shows that the mean-square error e_{agg} is smaller than the average e_{ave}. Notably, we hold the assumption that the performance of each classifier $\sigma(\boldsymbol{x}, \mathbf{A})$ on training set replicas is similar to the one on the entire training set. Moreover, the more diverse the simple predictor $\sigma(\boldsymbol{x}, \mathbf{A})$ is, the more accurate the composite results that the aggregated predictor will produce. For the CNNBagging architecture, as each deep CNN can be seen as an unstable weak classifier, the bagging strategy can boost the performance. However, the bootstrap ratio, or the drawn probability distribution P from $\mathbf{A}$, is also crucial for improving the aggregation.

4 | PERFORMANCE ANALYSIS

4.1 | Computational complexity

The computational complexities of the proposed algorithms are analyzed in this subsection. As described in subsections 3.2 and 3.3, the training process is implemented offline, which greatly improves efficiency. Despite the feedforward to output and weighted fusion in the reference stage, the computation burden is mainly determined by iteration number T and the complexity of each CNN. Assume that the dimensionality and number of training samples are N_1N_2 and L, respectively. For CNNBoosting, the computational complexity of feature extraction and logistic regression are $\mathrm{O}\left(N_1^2N_2^2L^3\right)\times I_T$ and $\mathrm{O}\left(N_1^2N_2^2L^2\right)$ for single CNN, where I_T denotes the iteration number of the optimization process. Then, the total complexity of CNNBoosting approximates to $I_T\mathrm{O}\left(N_1^2N_2^2L^3\right)+\mathrm{O}\left(N_1^2N_2^2L^2\right)$. However, the bootstrap ratio η indicates the balance between performance and complexity in terms of the number of training samples in CNNBagging. Similar to CNNBoosting, the entire computation of CNNBagging amounts to $I_TO\left(N_1^2N_2^2\eta^3L^3\right)+\mathrm{O}\left(N_1^2N_2^2\eta^2L^2\right)$

as the training number is reduced to ηL. It is clear that the computational cost is enormously large when the available samples increase sharply. However, the deep convolutional architecture is beneficial for extracting abstract invariant features. Moreover, ensemble methods help to collaboratively determine the final prediction in the fusion layer. As a result, these two techniques jointly improve their recognition performances accurately and effectively.

4.2 | Ensemble methods extension

Ensemble learning prevails in classification by linearly combining weak learners. Typical methods include boosting, bagging, random forest, and random subspace. To achieve the multi-platform fusion recognition of radar emitters, we focus on the crucial problem of non-negative weight assignment to form a composite decision. The update rule during iteration depends on the classification error incurred by each CNN. Once the weight coefficients $\{\beta_1, \ldots, \beta_T\}$ are produced, the leveraging combined decision is consequently formed. Therefore, it is advantageous to combine ensemble learning and CNNs. The proposed scheme not only automatically extracts deep features but also boosts the decision-level performance. For practical emitter recognition, it is helpful to combine the information of multiple heterogeneous sensors. Hence, ensemble methods are suitable for decision-making in our paper.

5 | EXPERIMENTS

In this section, extensive experiments are conducted to present the performance of the proposed algorithms. The experiments are implemented on full-pulse radar signals. Fusion recognition and results analyses are demonstrated.

5.1 | Simulation dataset and setup

To compare the performances, six types of radar signals are chosen as the training and testing samples, namely conventional pulse (CP), linear frequency modulation (LFM), nonlinear cosine frequency modulation (NCFM), binary phase shift keying (BPSK), binary frequency shift keying (BFSK), and quadrature frequency shift keying (QFSK). The experimental setup for the dataset is as follows. The sampling rate is 50 MSPS, and the carrier frequency is 5 MHz for all the above-mentioned types of signals. For BPSK and BFSK, we take the Barker code, while for QFSK, we take the Frank code. Training signals are generated in an SNR range of 20 dB to 30 dB. To evaluate the robustness to noise, testing samples with the same settings are generated with different SNRs. Every experiment is conducted for 1200 Monte Carlo simulations. The proposed ensemble CNNs are constructed as follows. Each CNN module is composed of 5 layers, the sizes of the convolutional and downsampling layers of which are 5×5 and 2×2, respectively. The number of CNNs T is selected accordingly.

To present the extracted features clearly, the deep features of emitter signals are visualized in Figure 2. Every 2 patches of Figure 2 in the column represent a certain type of feature. Each patch is produced by CNNs, and the size is 26×26. It reveals that different types of features show distinctive differences. Although the discriminations are not given in a mathematical form, the visualization provides an alternative.

5.2 | Parameters tuning

To demonstrate the performance, the case of two surveillance platforms is first considered. Although CNNBoosting and CNNBagging seek for the optimum weights in the pre-training stage with an ensemble strategy, the difference between training set $\mathbf{A}$ and the CNN model should be considered. The learning rates ω are randomly preset as $\{0.3, 1\}$, and the whitened training set is $\mathbf{A}^{64\times64\times1200}$. Then, ensemble weights are produced after several iterations and normalization, at $\{0.565, 0.435\}$ and $\{0.309, 0.691\}$, respectively. The ensemble learning method updates the weights according to the properties of CNNs, rather than assigning the same weights for β.

To investigate the effect of ω, we change it in fixed steps of 0.1 in the range $\omega \in [0, 1]$. The average recognition accuracy is used to measure the performance. The SNR of the testing sample is set to 4 dB. The experimental results are shown in Figure 3A and 3B. They demonstrate that a slightly moderate learning rate at approximately $\omega \in [0.4, 0.7]$ promotes an optimal performance. Too small or too big a value of ω will lead to a sharp decrease in recognition probability. In comparison, CNNBoosting outperforms CNNBagging in the entire accuracy range, especially for $\omega \in [0.4, 0.7]$.

In practice, the case in which the number of surveillance platforms exceeds 2, that is, $T \geq 2$, should be considered. To

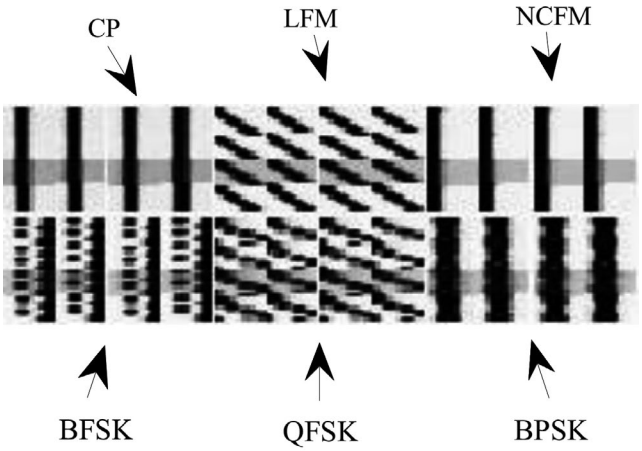

FIGURE 2 visualization of deep features

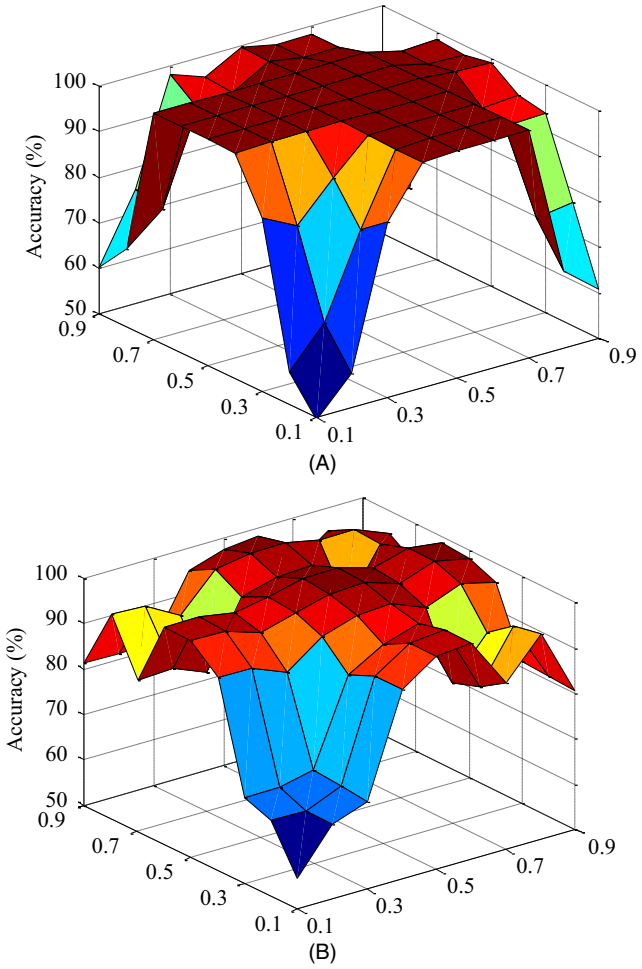

FIGURE 3 Tuning of learning rates ω: (A) ω tuning of CNNBoosting; (B) ω tuning of CNNBagging

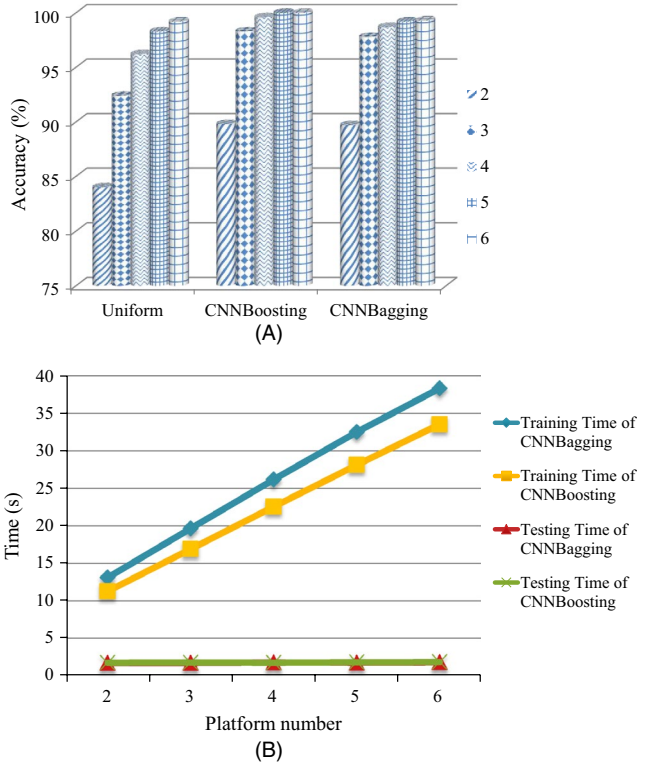

FIGURE 4 Tuning of platform number: (A) T tuning of CNNBoosting; (B) T tuning of CNNBagging

achieve an identical signal that is intercepted by multiple platforms, we keep the parameters the same as aforementioned except for T. To illustrate the importance of weight assignment, we take uniform weights for comparison when SNR = 0 dB. In particular, uniformity implies that the parameter settings of the CNNs and the weights are identical. The results are presented in Figures 4A and 4B, where Figures 4A and 4B show the relationship between T and accuracy and running time, respectively.

Figure 4A indicates that a higher number of platforms helps improve the recognition accuracies, which can be attributed to the increased information and feature measurements supplied by other platforms. With the increase in T, the accuracies of the proposed algorithms approach 100%. In comparison, the trained and selected weights in CNNBoosting and CNNBagging outperform the uniform one significantly. This may be because both the training process and ensemble learning help boost the fusion recognition.

However, although increasing the platform number helps improve accuracy, the computational complexity also increases. Figure 4B indicates the training and testing times. When T increases, the training time exhibits a nearly linear growth, and CNNBoosting consumes more time in comparison. In addition, the testing time does not display an obvious increase. This is because, once the proposed structures are pre-trained offline, the testing time mostly comprises the execution of forward-propagation with the well-trained models. Hence, the agility can be guaranteed in practice.

5.3 | Confusion matrix and robustness

To illustrate the effectiveness for the listed emitters, the confusion matrix is given in Table 2. In the experiments, testing signals are mixed with 0 dB of measured noise. As the confusion matrix shows, the diagonal elements are the right recognition accuracies, whereas the others are false. The bracketed elements are results of CNNBagging. The results in Table 1 show that the proposed algorithms are suitable for the recognition of emitter signals. For BPSK, BFSK, and QFSK, the right accuracies are not as high as those of the other types. This can be seen from the visualized features, where CP, LFM, and NCFM have distinguished features. However, all of them hold an accuracy of approximately 90% with an SNR = 0 dB. This indicates that the proposed algorithms have the ability to learn good features and obtain high accuracy.

TABLE 1 Confusion matrix of CNNBoosting at 0 dB

	Actual types					
Fusion inference	**CP**	**LFM**	**NCFM**	**BPSK**	**BFSK**	**QFSK**
CP	**94** (93.1)	0.4 (0.2)	0 (0.6)	0.7 (0.3)	1.0 (0.9)	2.5 (1.2)
LFM	0 (0.5)	**92.5** (92.4)	0.7 (0.2)	0.8 (1.0)	0.3 (3.6)	0.4 (3.0)
NCFM	1.6 (0)	2.3 (0.6)	90.4 (**91.6**)	3.2 (3.9)	3.8 (1.6)	1.4 (0.9)
BPSK	2.5 (2.6)	3.4 (3.1)	4.5 (2.8)	**87.6** (86)	4.2 (4.2)	2.7 (4.3)
BFSK	1.6 (3.0)	1.2 (2.2)	2.4 (4.4)	3.6 (3.8)	86.5 (**88.9**)	5.0 (4.6)
QFSK	0.3 (0.8)	0.2 (1.5)	2.0 (0.4)	4.1 (5.0)	4.2 (0.8)	**88** (86.0)

TABLE 2 Recognition accuracies of different feature extraction methods

Method	**−6**	**−4**	**−2**	**0**	**2**	**4**	**6**	**8**	**10**
B-MLP [21]	52.8	66.7	73.6	82.2	88.4	90.6	95.3	97.4	97.9
RPSC + En [4]	58.5	71.6	78.4	85.2	90.5	93.8	96.9	98.2	98.3
RP + KNN+En	55.2	68.6	77.5	85.2	91.2	95.4	97.3	98.3	98.3
RP + PSVM+En	56.3	70.4	79.3	86.8	92.2	96.2	98.0	98.6	98.6
TFI-CNN [11]	58.4	72.5	80.5	87.8	92.4	96.5	98.2	98.8	98.8
LWRT [12]	59.3	74.4	82.3	88.0	92.8	96.8	98.4	98.9	99.0
CNNBagging	60.1	75.0	82.9	88.8	93.2	97.2	98.6	99.2	99.3
CNNBoosting	63.4	77.2	84.2	88.9	93.5	98.1	99.4	99.5	99.5

5.4 | Comparisons

Because the proposed algorithms are aimed at extracting features automatically and performing decision-level fusion, this subsection presents comparisons with state-of-the-art algorithms for feature extraction and fusion recognition.

5.4.1 | Feature extraction

This subsection mainly compares feature extraction between shallow and DL methods. Presently, shallow methods, such as feature-based B-MLP (Bayesian MLP) [21] and compression-based RPSC [4], exhibit excellent performance. Furthermore, TFI-CNN [11] and LWRT [12] are taken as typical deep methods, as both adopt CNN as the feature extractor. For a fair comparison, the signal parameters are set identical, and the platform number is set to 2. We obtain sparse solution of the RPSC using the GRSR algorithm and extract 42 dimensional features in B-MLP. Furthermore, PSVM and KNN are taken as classifiers in RPSC to replace the SC classifier. Trained weights from CNNBoosting and CNNBagging are used in the fusion layer for the other methods (represented as En for short). TFI-CNN and LWRT both use uniform weights.

Table 2 indicates that deep methods show an obvious advantage over shallow methods in terms of recognition accuracy in the given SNR range. It proves to be more robust to noise, especially for low SNRs, which are inevitable in practice. This suggests that deep methods have a better capacity for learning and exploiting discriminative feature representations. Compared with TFI-CNN and LWRT, which directly use CNN, the proposed algorithms show higher recognition probabilities. This is because CNNBagging and CNNBoosting remove the delicate adjustment of CNN parameters and balance the relationship between CNN and weights.

5.4.2 | Fusion recognition

As the proposed ensemble CNNs achieve decision-making fusion, we compare them with other combination methods, including maximum, average, product, Bayesian, and Dempster-Shafer (D–S) fusion. For a fair comparison, the constructed CNNs are still used to exploit deep features, and

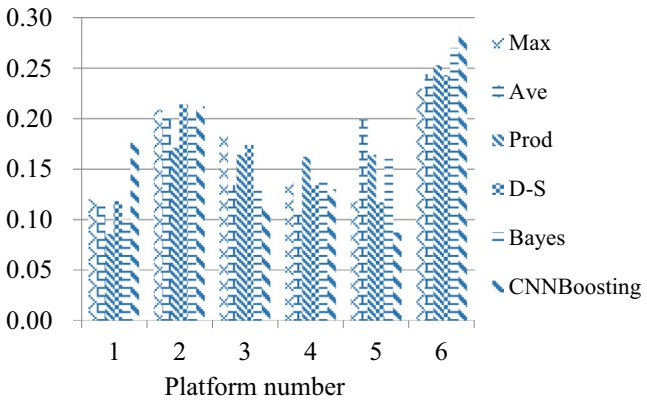

FIGURE 5 Probability distribution of different combinations

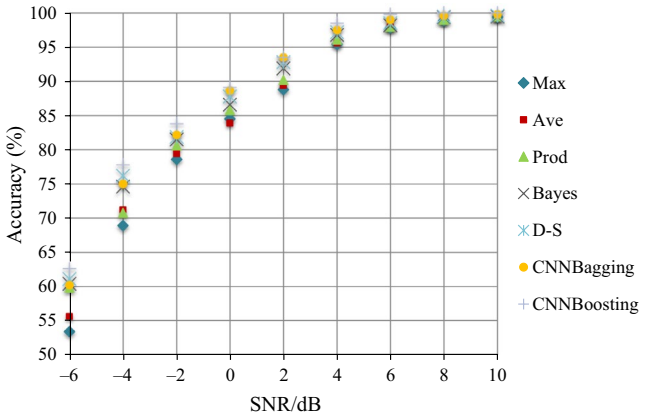

FIGURE 6 Recognition accuracies of different combinations

the parameter settings remain the same. Because the proposed models search for the optimum weights, the comparison methods are assigned with the same weights (0.5 for 2 sensors), and pre-training is performed without any special processing.

Figure 5 demonstrates the normalized fusion probability distribution. Here, 2 single inferences differ with an SNR = 0 dB, and the right judgment is $H(\mathbf{Y}_{fusion}) = 6$. All the combination methods are observed to yield the highest probability at label 6 and achieve right inference after integration. In the comparison, CNNBoosting ranks first, which implies that it yields the most reliable inference. This is owing to the better usage of the CNN's structure and the distribution of training data in the pre-training stage.

Further, the noise robustness of the combination is analyzed in Figure 6. Undoubtedly, the accuracy of CNNBoosting is the highest in the entire SNR range. However, the proposed CNNBagging exhibits worse robustness compared with Bayesian and D-S fusion when SNR < 0 dB. The performance of CNNBagging improves when SNR increases. It is noted that all the methods approximate a similar performance when SNR < 0 dB. In such cases, a high SNR contributes slightly. In summary, the proposed models and weight assignment method jointly improve the robustness of fusion recognition.

6 | CONCLUSIONS

In this paper, methods of integrating DL and ensemble learning for the automatic recognition of multi-platform radar emitters are investigated. In contrast to conventional algorithms, the proposed method constructs a more effective architecture for handling deep feature extraction and fusion recognition. Deep features that are produced by CNNs and fusion structures that are constructed using ensemble learning collaboratively boost the performance. Experiments conducted on six types of radar emitters demonstrate that the proposed algorithms outperform state-of-the-art feature extraction methods and conventional fusion techniques. To further develop the algorithms, research could be devoted to improving the structure of CNNs and ensemble methods. Although the proposed novel architecture is effective and robust, the hierarchical process takes a long time to learn deep features.

ACKNOWLEDGMENTS

The authors are grateful to the editor and anonymous reviewers for their valuable comments on this article. This study was co-supported by the National High Technology 863 Program (No. 2014AA7014061) and the National Natural Science Foundation of China (No. 61501484).

ORCID

Zhiwen Zhou https://orcid.org/0000-0001-5325-0669

REFERENCES

1. R.G. Wiley, *ELINT: The interception and analysis of radar signals*, Artech House Publishers, London, USA, 2006.
2. Q. Guo et al., *Recognition of radar emitter signals based on SVD and AF main ridge slice*, J. Commun. Netw. **17** (2015), no. 5, 491–498.
3. D. Zeng et al., *Automatic modulation classification of radar signals using the generalized time-frequency fepresentation of Zhao*, Atlas and Marks, IET Radar, Sonar Navig. **5** (2010), no. 4, 507–516.
4. J. Ma et al., *Robust radar waveform recognition algorithm based on random projections and sparse classification*, IET Radar Sonar Navig. **8** (2014), no. 4, 290–296.
5. C. Andre, S.L. Hegarat-Mascle, and R. Reynaud, *Evidential framework for data fusion in a multi-sensor surveillance system*, Eng. Appl. Artif. Intell. **43** (2015), 166–180.
6. Z.W. Zhou, G.M. Huang, and J. Gao, *An emitter fusion recognition algorithm based on multi-collaborative representation*, in Proc. Int. Cong. Image Signal Process, Liaoning, China, 2015, pp. 1231–1235.
7. C.X. Chen, M.H. He, and H.F. Li, *An improved radar emitter recognition method based on Dezert-Smarandache theory*, Chinese J. Electron. **24** (2015), no. 3, 611–615.
8. Y. Bengio, *Learning Deep Architectures for AI: Foundations and Trends in Machine Learning*, Now Publishers, USA, 2009.
9. J. Schmidhuber, *Deep learning in neural networks: an overview*, Neural Netw. **61** (2015), 85–117.
10. X.B. Wang et al., *Radar emitter recognition based on the short time fourier transform and convolutional neural networks*, in Proc. Int. Cong. Image Signal Process., BioMed. Eng. Inform., Shanghai, China, 2017, pp. 1–5.
11. C. Wang, J. Wang, and X.D. Zhang, *Automatic radar waveform recognition based on time-frequency analysis and convolutional neural network*, in Proc. Int. Conf. Acoust., Speech, Signal Process., New Orleans, LA, USA, 2017, pp. 2437–2441.
12. S.H. Kong et al., *Automatic LPI radar waveform recognition using CNN*, IEEE Access **6** (2018), 4207–4219.
13. M. Skurichina and R.P.W. Duin, *Bagging, boosting and the random subspace method for linear classifiers*, Pattern Anal. Appl. **5** (2002), no. 2, 121–135.

14. Y. Freund, *Boosting a weak learning algorithm by majority*, Inf. Comput. **121** (1995), no. 2, 256–285.
15. T. Thayaparan et al., *Time-frequency approach to radar detection, imaging, and classification*, IET Signal Process. **4** (2010), no. 4, 325–328.
16. R. Moradi and R. Yousefzadeh, *Recognizing objectionable images using convolutional neural nets*, in Proc. Signal Process. Intell. Sys. Conf., Tehran, Iran, 2015, pp. 133–137.
17. M. Anthimopoulos et al., *Lung pattern classification for interstitial lung diseases using a deep convolutional neural network*, IEEE Trans. Med. Imaging **35** (2016), no. 5, 1207–1216.
18. Y. Freund and R.E. Schapire, *A decision-theoretic generalization of on-line learning and an application to boosting*, Lecture Notes Comput. Sci. **55** (1999), 23–37.
19. L. Breiman, *Bagging predictors*, Mach. Learn. **24** (1996), no. 2, 123–140.
20. D.C. Tao et al., *Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval*, IEEE Trans. Pattern Anal. Mach. Intell. **28** (2006), no. 7, 1088–1099.
21. J. Lundén and V. Koivunen, *Automatic radar waveform recognition*, IEEE J. Sel. Top Signal Process. **1** (2007), no. 1, 124–136.

AUTHOR BIOGRAPHIES

Zhiwen Zhou received his BS degree in communication engineering from Xiamen University in 2011 and his MS and PhD degrees in communication and information systems in the Naval University of Engineering, Wuhan, China, in 2014 and 2017, respectively. Currently, he is a lecturer in command college of the PAP, Tianjin, china. His main research interests include machine learning, data fusion, and artificial intelligence.

Gaoming Huang received his BS and ME degrees in electronic warfare from the Naval Electronic College of Engineering, China, in 1995 and 1998, respectively, and his PhD degree in signal processing from Dongnan University, Nanjing, China, in 2006. He is currently the chief professor in the Department of Information Countermeasures, Naval University of Engineering, China. His research interests are passive detection, intelligent signal processing, and electronic warfare system simulations.

Xuebao Wang received his BS degree in Electronic and Information Engineering at Northeastern Forestry University, Harbin, China, in 2014 and his MS degree in Communication Engineering at the Naval University of Engineering, Wuhan, China, in 2016. Now, he is a doctoral candidate in the Naval University of Engineering, China. His main research interests are radar signal processing and electronic countermeasures.