A Lightweight Decentralized Learning-Based Automatic Modulation Classification Method for Resource-Constrained Edge Devices

Biao Dong, *Student Member, IEEE*, Yuchao Liu, Guan Gui, *Senior Member, IEEE*, Xue Fu, Heng Dong, Bamidele Adebisi, *Senior Member, IEEE*, Haris Gacanin, *Fellow, IEEE*, and Hikmet Sari, *Life Fellow, IEEE*

***Abstract*—Due to the computing capability and memory limitations, it is difficult to apply the traditional deep learning (DL) models to the edge devices (EDs) for realizing lightweight automatic modulation classification (AMC). Recently, many works attempt to use different ways to realize lightweight AMC methods for edge devices (EDs). However, the lightweight seems to be a contradiction with the classification performance in these lightweight networks. In this paper, we propose an efficient lightweight decentralized learning-based automatic modulation classification (DecentAMC) method using spatio-temporal hybrid deep neural network based on multi-channels and multi-function blocks (MCMBNN). Specifically, the lightweight network is designed from the perspectives of comprehensive consideration of lightweight and classification performance, which is composed of three parts to extract different features for realizing high classification performance and they are parameter estimator and transformer (PET) block, spatial information extraction block and temporal feature extraction & Softmax block. In addition, we use multi-channel input to extract complementary features of different channels for a better classification performance. The proposed DecentAMC method is an efficient training method, which is achieved by the cooperation in which multiple EDs update and upload the model weight to a central device (CD) for model aggregation to avoid the data privacy disclosure and reduce the computing power and storage pressure of CD. Experimental results show that the proposed MCMBNN can obtain an improved classification accuracy while reducing model complexity with the contributions of three blocks. Moreover, the proposed DecentAMC method can be deployed on EDs efficiently. Thus, the method has the advantages of avoiding data leakage on EDs and relieving the computing pressure of CD with relatively lower communication overhead. The simulation code and datasets are shared on GitHub.**[1]

***Index Terms*—Automatic modulation classification, lightweight neural network, decentralized learning, spatio-temporal hybrid deep neural network, Internet of Things.**

This work was supported by the National Key Research and Development Program of China under Grant 2021ZD0113003, the Key Project of Natural Science Foundation of the Higher Education Institutions of Jiangsu Province under Grant 22KJA510002. *(Corresponding author: Guan Gui).*

Biao Dong, Guan Gui, Xue Fu, Heng Dong, and Hikmet Sari are with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: 1220013135@njupt.edu.cn, guiguan@njupt.edu.cn, 1020010415@njupt.edu.cn, dongh@njupt.edu.cn, hsari@ieee.org).

Yuchao Liu is with the China Research Institute of Radiowave Propagation, Qingdao 266107, China (e-mail: lyc8541832@163.com).

Bamidele Adebisi is with the Department of Engineering, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom (e-mail: b.adebisi@mmu.ac.uk).

Haris Gacanin is with the Institute for Communication Technologies and Embedded Systems, RWTH Aachen University, Aachen 52062, Germany (e-mail: harisg@ice.rwth-aachen.de).

[1]The code of this manuscript can be obtained from the following link: https://github.com/dongbiao321/MCMBNN-for-CentAMC-and-DecentAMC

I. Introduction

A. Background

The internet of things (IoT) is an information service paradigm, which defines an interrelated dynamic environment for the integration of terminal devices and realizes seamless connection of devices and data transmission [1]–[3]. With the increasing number of terminal devices, massive amounts of data are produced in IoT devices and generally stored in untrusted storage [4]. In addition, IoT devices are vulnerable to external malicious attacks [5]–[7]. Hence, they are at a risk of data leakage when devices can not recognize the external malicious attacks. Automatic modulation classification (AMC) is a promising technique that can be applied at the receiver to distinguish the different types of modulated signals [8], which is an important way to identify malicious attacks from the physical layer like spectrum sensing data falsification (SSDF) attacks and jamming attacks [9]. Recently, AMC has played a significant role in both military and civilian communications, such as cognitive radio and link adaptation.

In the past decades, typical AMC methods included likelihood-based AMC (LBAMC) [10]–[15] and feature-based AMC (FBAMC) [16]–[22]. The LBAMC method models the classification problem as a composite hypothesis and uses the likelihood function to determine the correct type of modulation. Generally, on the basis of different likelihood functions, LBAMC methods mainly include average likelihood ratio test (ALRT) [10], [11], generalized likelihood ratio test (GLRT) [12], [13] and hybrid likelihood ratio test (HLRT) [14], [15]. Although LBAMC can achieve the optimal performance in a Bayesian sense, it requires a high computational complexity and hence it is difficult to apply in a real system without channel state information.

Different from the LBAMC method, extraction of specific features from the received signal and completion of classification tasks are the approaches used in the FBAMC method, which can process signals with unknown channel statistical characteristics and has less computational complexity than the LBAMC method [16]. Existing FBAMC methods generally extract statistical features, such as cyclic statistics [17], [18], wavelet transform [20] and cumulants [19] for final decision making. Traditional machine learning (ML) methods, such as decision tree [21] and k-nearest neighbor (KNN)

[22], were also widely used in the FBAMC methods as a classifier. However, these traditional methods need to extract the features manually and the classification performance is poor in processing multiple signals classification task.

B. Motivations

In recent years, deep learning (DL) has made great achievements in many fields, such as wireless communications [23]–[26], IoT devices attack detection [27], [28] and IoT malware classification [29]. The advantages of DL methods compared with traditional machine learning methods can be summarized as follows: first, DL methods can realize feature learning internally without special feature extraction, which avoids complex feature engineering based on expert knowledge [30]. Second, more advanced features can be extracted through the combination of different DL layers for achieving better performance. Hence, more and more scholars tried to apply DL methods to the FBAMC method for improving classification accuracy [30]–[47].

1) Neural network design for AMC methods: Most of current methods based on DL are directly borrowed from the field of image processing or natural language processing [30], [34]–[40], [44]. While there are few networks specially designed for the purpose of signal modulation. Hence, to further improve the classification accuracy of AMC, it is necessary to design the network in accordance with the characteristics of the modulation signals in order to facilitate learning the signal features from multiple perspectives. For instance, Lin *et al.* adopted short-time Fourier transform (STFT) to transform the modulated signals into spectrum, and extracted richer features of the modulated signals from the perspective of time-frequency analysis [31]. Chang *et al.* adopted the in-phase/quadrature (I/Q) format and amplitude/phase (A/P) format in different SNR, and fused the features of the two signals for better Classification performance [32].

2) Lightweight neural network for DecentAMC: Existing DL-based AMC methods generally use the algorithm of local learning (LocalAMC) or centralized learning (CentAMC) with convolution neural network (CNN) or deep residual network (ResNet). The LocalAMC means that each edge device (ED) relies on a limited local dataset for training without using datasets from other EDs, which leads to a limited performance. The CentAMC means that multiple EDs upload the local datasets to a central device (CD) for training, which has a better performance because of training the model with multiple datasets, as is shown in Fig. 1. However, CentAMC challenges the computing capability and storage of CD and threatens the privacy security of data during the process of dataset sharing.

To solve the problems of LocalAMC and CentAMC, a distributed learning-based AMC (DistAMC) method based on CNN was proposed in [33], which can realize decentralized training of data by the way of multiple devices uploading the model weight rather than datasets to a CD. However, the model size of CNN adopted in DistAMC is large and the model weight needs to be updated frequently, which may lead to huge communication cost. In addition, CNN has a high complexity and thus is hard to deploy to EDs. Hence, we attempt to deploy lightweight network specifically designed for modulated signals in the decentralized learning-based (DecentAMC) method.

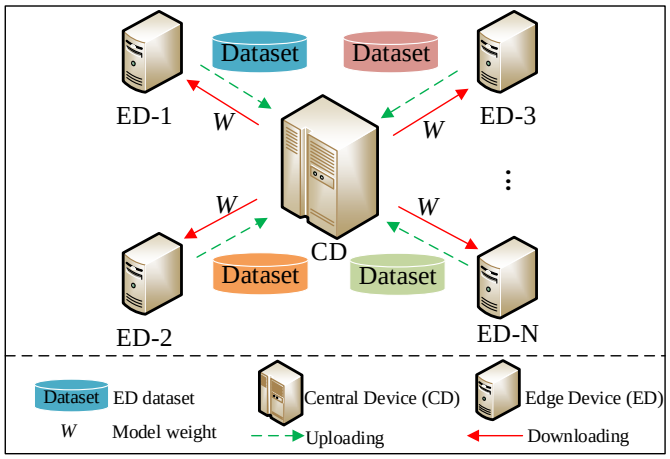


Fig. 1. The structure of the traditional CentAMC method. Traditional edge devices generally adopt CentAMC methods. Multiple edge devices (EDs) upload the local datasets to a central device (CD), and then the CD trains the model based on the aggregated datasets. When the training is finished, the model weight W of the CD is downloaded to the EDs for testing.

C. Related Works About DL-based AMC Methods

We present the related DL-based AMC works in two groups. First are high-performance networks, whose contributions are mainly about achieving a high classification performance through combining different network layers or deepening the network depth. Second are lightweight networks, which aims to reduce the complexity of the network and make it easier to deploy on EDs.

1) High-performance networks: An earlier DL-based AMC method is convolution neural network based AMC (CNN-AMC) [30], which can outperform the FBAMC and has a faster computing speed with parallel computation. Zhang *et al*. proposed an improved CNN-AMC network through fusing images and handcrafted features of signals for extracting more advanced features, which can improve classification accuracy compared with previous works [34]. Another optimized CNN-AMC network named SBCNN also has a high robustness on a complex dataset by designing an optimal filter size for improving the prediction accuracy [36]. Lin *et al* designed a time-frequency attention mechanism for CNN-AMC to learn which frequency, channel and time features are more meaningful in networks for modulation classification [31], which outperforms other attention mechanisms. In [37], multiple CNN models are trained for multi-task learning under different SNRs and each CNN model shares the model weight with other CNN models, which can be applied in realistic noise scenarios and achieves better generalization and robustness. There are also numerous excellent networks in the field of image processing that have been applied to AMC besides CNN. P. Qi *et al*. addressed methods to realize deep residual networks (ResNet)-based AMC method [40], which can efficiently distinguish among sixteen modulated signals.

A dual path network (DPN) consisted of multiple residual blocks is proposed in [35], which avoids gradient extinction while deepening the network and outperforms other signal processing algorithms at SNRs above 14 dB.

In addition, modulated signals have not only spatial characteristics, but also temporal characteristics. Hence, a CNN-LSTM network is proposed in [44], which extracts spatial features with CNN and extracts temporal features with LSTM to complete the fusion of temporal and spatial features. Another CNN-LSTM network based multi-channel learning framework is proposed in [45], which obtains better recognition accuracy especially for higher dimensional schemes (16-QAM and 64-QAM) and has faster convergence speed compared with other state-of-the-art DL networks,

2) Lightweight networks: Many scholars have adopted different ways to realize lightweight networks. A group-level sparsity based lightweight network for AMC is proposed in [38], which can make model pruning itself to achieve a compact network and solve the problem of recognition confusing types with a improved two-step training method. F. Teng *et al* proposed a channel compensation mechanism and an accumulated polar feature-based DL method, which can reduce training overhead under time-varying fading channels [49]. A novel AMC method based on neural architecture search (NAS) is given in [39], which can automatically design the high-performance network structure with lower model complexity. S. Luan *et al* applied shuffle unit and Gated Recurrent Unit (GRU) to make the network more lightweight for solving the time-consuming problem under impulsive noise [50]. Y. Wang *et al.* [41] proposed a lightweight network for AMC using the combination of DL and compressive sensing, which made CNN become lightweight with a slight performance loss. Y. Lin *et al.* [42] proposed an improved lightweight AMC method by considering the pruning technology to reduce the size of neural networks for promising edge applications. A lightweight neural network based on separable convolution (SCNN) for AMC was proposed in [47], which mainly adopts separable convolutions to replace parts of standard convolution of CNN and parts of FC layers of CNN are removed. The model complexity of SCNN was decreased by about 94% when compared with CNN. An optimized CNN network MCNet is proposed in [43], which is concatenated or added by multiple convolution blocks with asymmetric convolution kernel. MCNet can effectively capture the spatial correlation of modulation signals with the increase of the number of convolution blocks. Another lightweight network based on phase parameter estimation and transformation (PET), with CNN lay and GRU lay as the feature extraction layers of space and time, which can reduce a third of the volume of parameters with a slight performance loss compared with the existing state-of-the-art networks [46].

D. Main Contributions

In this paper, an efficient lightweight neural network named spatio-temporal hybrid deep neural network based on multi-channels and multi-function blocks (MCMBNN) for DecentAMC method is proposed to solve the problem of insufficient computing power and storage limitation of EDs when applying the traditional DL model to EDs for realizing AMC. The main contributions of this paper include:

- A novel MCMBNN specially designed for modulated signal classification is proposed, which adopts three channels to extract features of the modulation signals and finally realizes the feature fusion of different signal channels. Meanwhile, MCMBNN can extract the phase feature, spatial feature and temporal feature of the modulation signals with low model complexity by specially designed function blocks. Experimental results show that MCMBNN has a high robustness on the different datasets.
- An Efficient training method named DecentAMC method is proposed, which can avoid the data privacy disclosure and relieving the computing pressure of CD with the cooperation in which multiple EDs update and upload model weight to a CD for model aggregation.
- We train the MCMBNN with the DecentAMC method and the experimental results show that the proposed MCMBNN based DecentAMC method can be deployed on EDs efficiently, which has the advantage of lower communication cost.

TABLE I
THE SUMMARY OF ABBREVIATIONS IN THIS PAPER.

Abbreviations	Full Name
A/P	Amplitude/Phase
Adma	Adaptive Moment Estimation
ALRT	Average Likelihood Ratio Test
AMC	Automatic Modulation Classification
CD	Central Device
CentAMC	Centralized Learning-Based Automatic Modulation Classification
CNN	Convolution Neural Network
DecentAMC	Decentralized Learning-Based Automatic Modulation Classification
DL	Deeping Learning
DSB	Double Sideband Modulation
ED	Edge Devices
FBAMC	Feature-Based Automatic Modulation Classification
FSK	Frequency-Shif Keying
GLRT	Generalized Likelihood Ratio Test
GRU	Gated Recurrent Unit
HLRT	Hybrid Likelihood Ratio Test
I/Q	In-phase/Quadrature
IoT	Internet of Things
KNN	K-Nearest Neighbor
LBAMC	Likelihood-Based Automatic Modulation Classification
LocalAMC	Local Learning-Based Automatic Modulation Classification
LSTM	Long Short-Term Memory
ML	Machine Learning
PET	Phase Parameter Estimation and Transformation
PSK	Phase Shift Keying
QAM	Quadracture Amplitude Modulation
ResNet	Deep Residual Network
SNR	Signal to Noise Ratio
SoA	State-of-the-Art
SSB	Single-Sideband Modulation
MCMBNN	Spatio-Temporal Hybrid Deep Neural Network Based on Multi-channels and Multi-function Blocks
WBFM	Wide Band Frequency Modulation

This paper is organized as follows. Section II depicts the signal model and the framework of DL-based AMC Method. Section III describes the designing process of lightweight neural network (MCMBNN) for AMC in detail. The DecentAMC as a training method is introduced by contrasting with CentAMC method in Section IV. Next, Section V presents the analysis of experimental results. Finally, the whole paper is concluded in Section VI. The summary of abbreviations in the paper is shown in Table I.

II. Signal Model and DL-based AMC Methods

A. Signal Model

We assume that the model of the unknown single-carrier (SC) modulated signals as

$$u(k) = \lambda e^{j(2\pi f_0 k/K+\theta)} \sum_{l=0}^{L-1} h[l]q[k-l-\tau]_{\mathrm{mod}K} + \sigma(k),\ k \in \{0,1,\cdots,K-1\}, \quad (1)$$

where $u(k)$ represents the unknown modulated signals at the receiver, λ represents the channel gain, f_0 the frequency offset, θ the phase offset, τ the timing offset, $h[l : l = 0, 1, \cdots, L-1]$ the channel impulse response (CIR) of Rayleigh fading channel, L is the length of the CIR, $q(k)$ is the baseband signal sequence, $\sigma(k)$ is additive white Gaussian noise, and K is the number of sampled points from the signals.

The in-phase component (I) and quadrature component (Q) of $u(k)$ called IQ signals are input to neural network for modulation classification and expressed as

$$I = \{\mathrm{real}[u(k)]\}_{k=0}^{K-1},\ \text{and}\ Q = \{\mathrm{imag}[u(k)]\}_{k=0}^{K-1}. \quad (2)$$

B. DL-based AMC Methods

Next we describe DL-based AMC method, which belongs to FBAMC and can complete feature extraction and classification simultaneously. The modulation type of the received modulated signal belongs to the set $D = \{d_j, j = 0, 1, \cdots, J-1\}$, where J represents the number of modulation types. DL-based AMC can be expressed as: $d_j = F_{d_j \in D}([I;Q], P)$, where $F(\cdot)$ is the function of DL-based AMC classifier, and P represents the parameter of the network. The location of DL-based AMC method in communication system is shown in Fig. 2

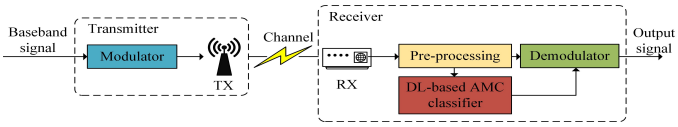


Fig. 2. The location of DL-based AMC method in communication system.

We adopt cross entropy (CE) loss function with ℓ_2 regularization to compile the model, which is expressed as

$$\mathscr{L}_{CE} = -\frac{1}{R}\sum_{r=1}^{R} y_r \log[F([I;Q],P)] + \lambda \mathscr{Q}(F([I;Q],P)), \quad (3)$$

where R represents the size of the training samples, and y_r is the true label, and $\mathscr{Q}(\cdot)$ is a penalty function to avoid overfitting, and λ is to balance the penalty function.

III. The Proposed Lightweight Neural Network for AMC

In this section, a novel lightweight neural network named MCMBNN is described elaborately. As shown in Fig. 3, the lightweight network MCMBNN is composed of multiple blocks with specific functions, i.e., a GRU layer and a dense layer with Softmax. According to the specific functions of different network layer, MCMBNN can be divided into three parts: Multi-channel Parameter Estimator and Transformer Block (Part A), Multi-channel Spatial Information Extraction Block (Part B), Temporal Feature Extraction & Softmax Block (Part C).

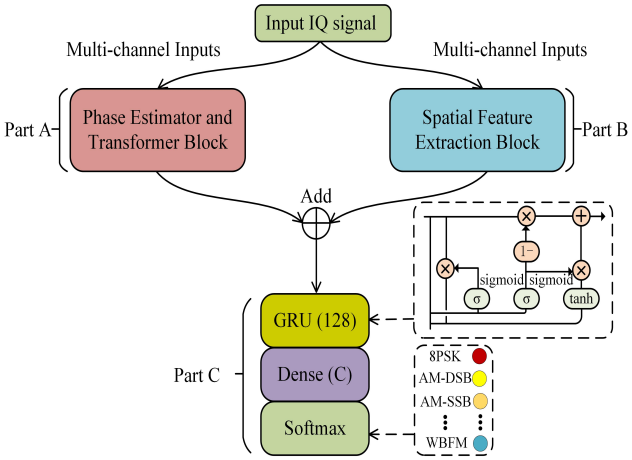


Fig. 3. The overall structures of the proposed MCMBNN.

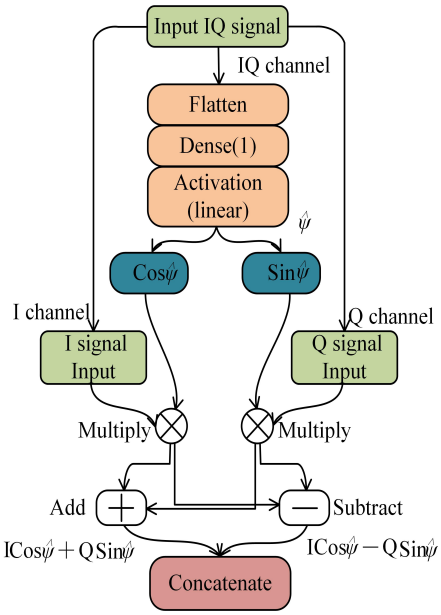


Fig. 4. The overall structures of PET.

A. Multi-Channel Learning

Original IQ channel signals are fed to Part A and Part B in parallel. Then, the input signals are divided into three streams: original IQ signal, I-channel signal and Q-channel signal,

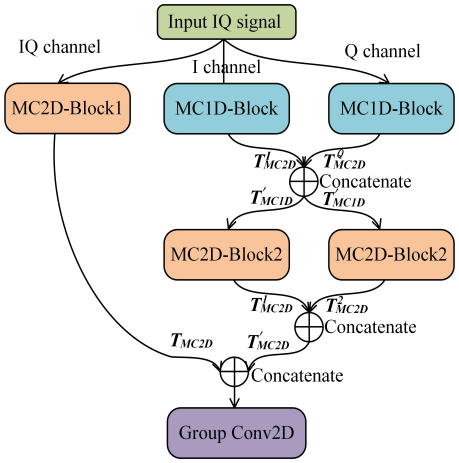
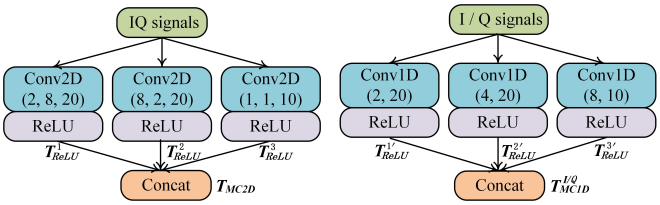


Fig. 5. The overall structures of spatial information extraction.



(a) MC2D-Block (b) MC1D-Block

Fig. 6. Two convolution blocks: MC2D-Block and MC1D-Block.

which is inspired by the multi-channel learning framework [45]. The reason for dividing the signal is that there exits great differences between the I-channel signal and the Q-channel signal and the features extracted by multiple channels can form a complementary relationship.

B. Parameter Estimator and Transformer Block

Original IQ channel signals usually carry the phase offset information due to the impact of channel noise and attenuation. Hence, phase estimator and transformer (PET) block can be applied in processing original signals for AMC to extract the phase offset information and thus improve the overall classification accuracy [46]. As shown in Fig. 4, phase estimator block is composed of a Flatten layer, a Dense layer with only one unit and a linear Activation function layer. Original IQ channel signals achieve dimension transformation through Flatten layer from the dimension (2,128) to a vector with dimension (1,256) for meeting the input dimension of Dense layer. And then, This vector can obtain rich phase feature information through the Dense layer. Finally, we use linear activation function to obtain an estimated phase parameters $\hat{\psi}$.

Phase estimator block is followed by phase transformer block, which can realize parametric inverse transformation.

$$\hat{u}[k] = u[k]e^{-j\hat{\psi}} = \begin{bmatrix} I\cos\hat{\psi} + Q\sin\hat{\psi} \\ I\cos\hat{\psi} - Q\sin\hat{\psi} \end{bmatrix}, \tag{4}$$

where $\hat{u}[k]$ is the output of the phase estimator block. In summary, the number of learned parameters in PET is relatively small and mainly concentrated in the Dense layer and Flatten layer.

C. Spatial Information Extraction Block

As shown in Fig. 5, three IQ channel signals are input to two different convolution blocks in parallel for extracting more advanced spatial features from three channels of signals. The first IQ mixed channel signal is input to MC2D-Block1, which is presented in Fig. 6(a). MC2D-Block is designed with three parallel two-dimensional standard convolution layer with two asymmetric convolution kernels ((2, 8), (8, 2)) and one (1,1) convolution kernel. Similar to [43], deploying asymmetric convolution kernel is to extract the signal spatial features in the horizontal and vertical dimensions without affecting the quality of spatial feature extraction, which can reduce the training parameters compared with the traditional symmetric convolution kernel ((8, 8) convolution kernel). And then, ReLU layer is followed by each convolution layer to enhance the nonlinearity of the network and prevent the gradient from disappearing, which adopts function $y = max(0, x)$. Considering that the output tensors of the ReLU layers as $T_{ReLU} \in \mathbb{R}^{H\times W\times C}$, where H and W represent the vertical and the horizontal dimension, respectively, and C denotes channel dimension. Later, the refined tensor $T_{MC2D} \in \mathbb{R}^{H\times W\times C}$ can be generated by concatenating the three output tensors of the ReLU layers in the channel dimension, which can be presented as: $T_{MC2D} = [T^1_{ReLU}; T^2_{ReLU}; T^3_{ReLU}]$.

The second I-channel input signal and the third Q-channel input signal are input to MC1D-Block respectively, as shown in Fig. 6(b). The overall structure of MC1D-Block is similar to MC2D-Block1. The difference is that MC1D-Block adopts one-dimensional convolution layer, because I-channel and Q-channel input signals are two one-dimensional sequences. Besides, three convolution layers use convolution kernels of different sizes (2, 4 and 8 respectively) to extract more comprehensive spatial features with less model training parameters. Finally, the refined tensor $T^{I/Q}_{MC1D} = [T^{1'}_{ReLU}, T^{2'}_{ReLU}, T^{3'}_{ReLU}]$, which means the output tensor of MC1D-Block is concatenated by three output tensors of the ReLU layers in the horizontal dimension.

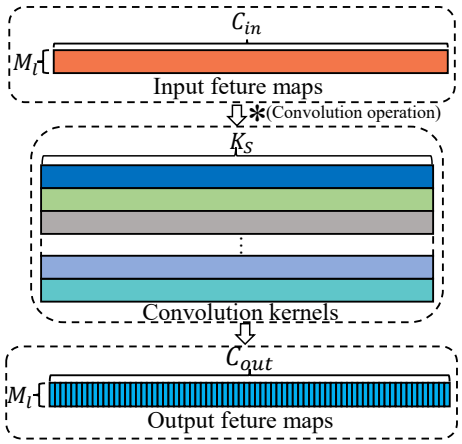
Next, two MC1D-Block output streams are concatenated in the vertical dimension to form a new tensor T'_{MC1D} , which fuses the spatial characteristics of I-channel and Q-channel.

$$T'_{MC1D} = \begin{bmatrix} T^I_{MC1D} \\ T^Q_{MC1D} \end{bmatrix}, \tag{5}$$

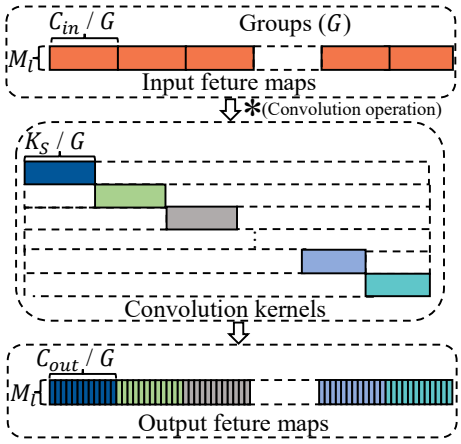
Then, T'_{MC1D} is fed into the MC2D-Block2 in parallel to further extract more abstract features. Compared with MC2D-Block1, the only difference with MC2D-Block2 is that MC2D-Block2 adopts (1,8) and (8,1) asymmetric convolution kernels. Later, two ouput tensors of MC2D-Block2 are concatenated in the channel dimension $T'_{MC2D} = [T^1_{MC2D}; T^2_{MC2D}]$. Next, T'_{MC2D} continues to fuse with T_{MC2D} in the channel dimension and the fused new tensor is fed to Group Conv2D layer.

From the perspective of reducing the complexity of the model, we use group convolution layer with (3,3) convolution

kernels and 2 group number instead of standard convolution layer with the same convolution kernels. Group convolution was first applied in AlexNet for training on multiple GPUs because of the limitation of single GPU [51]. The comparison



(a) Standard convolution



(b) Group convolution

Fig. 7. Comparison of standard convolution and group convolution. C_{in}, C_{out}, M_l and K_s respectively represent the number of input channels, and the number of output channels, and the length of out feature maps of convolutions, and the size of filters. G is the number of groups for group convolution. (a) standard convolution: the convolution kernels are conducted on all of the input channels. (b) group convolution: the input channels and the convolution kernels are equally divided into G groups when the number of groups is not 1.

of group convolution and standard convolution is shown in Fig. 7. The parameters and out feature maps comparisons of group convolution and standard convolution [52] can be calculated by

$$\frac{S^p_{group}}{S^p_{sta}} = \frac{K_s \cdot C_{in} \cdot C_{out} \cdot \frac{1}{G}}{K_s \cdot C_{in} \cdot C_{out}} = \frac{1}{G}, \tag{6}$$

$$\frac{S^g_{group}}{S^g_{sta}} = \frac{M_l \cdot C_{out}}{M_l \cdot C_{out}} = 1, \tag{7}$$

where S^p_{sta} and S^g_{sta} represent parameters and out feature maps of standard convolution respectively; S^p_{group} and S^g_{group} represent parameters and out feature maps of group convolution respectively.

It is obvious that the parameters of standard convolution are G times than that of group convolution, and the out feature maps are the same with group convolution by observing Eqs. (6)∼(7). It means that group convolution can generate the same size out feature maps with a smaller number of parameters. This is the theoretical basis for designing lightweight network using group convolution to take place of standard convolution.

D. Temporal Feature Extraction & Softmax Block

We use one Add layer for achieving the fusion of phase offset features and signal spatial features. Then, one GRU layer with 128 units is used to extract temporal features. Compared with the LSTM layer, GRU layer has a simpler network structure and can also extract temporal features efficiently. Finally, one Dense layer with C units (C is the number of classes of modulated signals) realizes the classification of the final modulated signal and the activation function is softmax.

As described above, the final MCMBNN has three parts, that adopt three different strategies to design the lightweight neural network. Firstly, PET block is applied in processing original signals to extract the phase offset features. Secondly, multiple groups of asymmetric convolution blocks and group convolution are used to extract spatial features. Thirdly, one GRU layer is used to extract temporal features. Meanwhile, in order to ensure the classification performance of the network, we adopt multi-channel input to extract the complementary features between different channels.

IV. The Proposed Training Method for AMC

In this Section, we first introduce the traditional CentAMC method in detail. On this basis, we propose the DecentAMC method.

A. The Traditional CentAMC Method

As shown in Fig. 1, each edge device (ED) has a local dataset $\mathcal{M}_n = \{(X_1, Y_1), (X_2, Y_2), \cdots, (X_{S_n}, Y_{S_n})\}$, where n represents the n-th ED and S_n represents the dataset size of the n-th ED. Hence, the global dataset can be denoted as $\mathcal{M}_g = \mathcal{M}_1 \cup \mathcal{M}_2 \cup \cdots \cup \mathcal{M}_N$, and the global dataset size can be denoted as $S_{global} = \sum_{n=1}^{N} S_n$. We assume that $\mathcal{M}_i \cap \mathcal{M}_j = 0$ for $i \neq j$.

In the CentAMC Method, the training process is performed on single edge device (ED) and is based on the global dataset collected from EDs. The global model is trained with the criterion of minimizing the empirical loss function, namely

$$\mathscr{F} = \frac{1}{S_n} \sum_{r=1}^{S_n} \mathscr{L}_{CE}, \tag{8}$$

Adaptive moment estimation (Adma) optimization is adopted to update the model weight ω for minimizing the empirical loss function

$$\omega_t = \omega_{t-1} - \eta_t \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{9}$$

where ω_t is the updated model weight of the t-th training epoch, $\hat{m}_t$ is the updated biased-corrected first moment estimate, $\hat{v}_t$ is the updated biased-corrected second row moment estimate, and η_t is the learning rate.

B. The Proposed DecentAMC Method

As shown in Fig. 8, compared with CentAMC method, each ED uploads the model weight instead of the dataset to the CD in DecentAMC method. It needs to be pointed out that the DecentAMC method falls within Federated Learning (FL) [55], which applies the idea of FL to model training process for AMC. The DecentAMC method consists of the following 4 steps, shown in **Algorithm 1**.

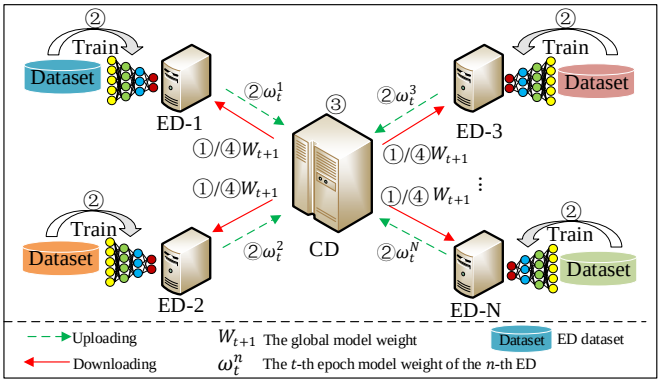


Fig. 8. The structure of the proposed DecentAMC method.

1) Model initialization and parameter broadcasting: The CD builds a model and initialization parameters, such as the initial model weight ω_0, the initial learning rate η_0, the number of EDs N, IQ samples and corresponding labels in the n-th ED dataset, the training epochs of global model T, and the batch size of a training epoch B. Then, the CD broadcasts the built model and initialized parameters to the EDs.

2) The model weight of the EDs updating and uploading: Each ED downloads the built model and initialized parameters from the CD, and then updates the local model weight based on the local datasets by Adam

$$\widetilde{\omega}_t^n = \widetilde{\omega}_{t-1}^n - \eta_t \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}, \tag{10}$$

When the updating is finished, each ED uploads its model weight to the CD. Here, we assume that the EDs upload the model weight $\widetilde{\omega}_t^n$ once every training epoch, where the t-th epoch model weight of the n-th ED as $\widetilde{\omega}_t^n$.

3) The EDs model weight aggregation by CD: The trained model weight is uploaded from each ED to the CD and the CD averages the received model weights. Model weight averaging is used as

$$W_{t+1} = \frac{\sum_{n=1}^{N} S_n \widetilde{\omega}_t^n}{S_{global}}, \tag{11}$$

where W_{t+1} is the t-th epoch global model weight.

4) The global model weight updating: After the CD gets the global model weight, each ED downloads the global model weight from the CD and replaces the original weight with the global model weight, i.e., $\widetilde{\omega}_{t+1}^n = W_{t+1}, n = [1, N]$ and then repeat 2) $\sim$ 4) until the loss convergence.

Algorithm 1: The proposed DecentAMC method.

Input: The number of EDs N; IQ samples and corresponding labels in the n-th ED dataset; the initial model weight ω_0; the initial learning rate η_0; the training epochs of global model T; the batch size of a training epoch B.

Output: W_T

CD initializes a model and broadcasts the model and initial global model weight ω_0 to EDs.
foreach *ED in parallel* **do**
  ED downloads the built model and ω_0.
  ED initializes parameter T, B, η_0.
end
for $t = 0, \cdots, T$ **do**
  $\widetilde{\omega}_t^n \leftarrow W_t$
  $\mathcal{M}_N \leftarrow$ Datasets of N EDs
  for *each ED* $\mathcal{M} \in \mathcal{M}_n$ *in parallel* **do**
    Each ED updates the local model weight.
    $\widetilde{\omega}_t^n \leftarrow \widetilde{\omega}_{t-1}^n - \eta_t \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}}$
    Each ED uploads their updated model weight.
  end
  CD updates global model weight
  $W_{t+1} \leftarrow \frac{\sum_{n=1}^{N} S_n \widetilde{\omega}_t^n}{S_{global}}$.
end
return W_T

V. Experimental Results

In this section, we conduct two experiments to evaluate the proposed lightweight neural network (MCMBNN) and the proposed training method (the DecentAMC method) based on three datasets. Firstly, we compare the MCMBNN with six SoA networks based on CentAMC, which include three high-performance networks CNN [30], [33], MCLDNN [45], CNN-LSTM [44] and three lightweight networks SCNN [47], MCNet [43] and PET-CGDNN [46]. MCLDNN and PET-CGDNN use the multi-channel learning framework while other networks choose to input IQ signals directly. MCLDNN, CNN-LSTM and PET-CGDNN contain temporal feature extraction units like LSTM or GRU, while CNN, MCNet and SCNN only consist of spatial feature extraction units. On this basis, we conduct ablation study experiment to explore the impact of each block of the proposed MCMBNN in terms of model complexity and classification performance. In the second experiment, we combine different networks with the DecentAMC method to achieve decentralized learning based on three datasets.

A. Dataset and Simulation Setting

1) Dataset: Three datasets are adopted to test the generalization performance of the model: RadioML.2016.10A, RadioML.2016.10B [48], and Dataset generated by Matlab [47]. RadioML.2016.10A and RadioML.2016.10B are standard open datasets, which are based on GNURadio. The difference of the data generated by GNURadio and MATLAB is that the IQ signals generated by GNURadio are constrained

to follow an analog baseband, while the MATLAB IQ points make discrete jumps [56]. Hence, simulations based on these two types of datasets can show the performance comprehensively. The signal length of each sample is 128, and the size of the input signal sample is $(2, 128)$ because the signal is input to the neural network in the form of IQ samples. The type of modulated signal of above three datasets as follows.

- RadioML.2016.10A: {BPSK, 8PSK, CPFSK, GFSK, PAM4, 16QAM, 64QAM, QPSK,AM-DSB, AM-SSB, WBFM}, the SNR of ranging from -20 dB to 18 dB with 2 dB as interval is used. The number of samples is 220,000.
- RadioML.2016.10B: {BPSK, 8PSK, CPFSK, GFSK, PAM4, 16QAM, 64QAM, QPSK,AM-DSB, WBFM}, the dataset of RadioML.2016.10B is a larger version RadioML.2016.10B of RadioML.2016.10A, which contains 1,200,000 samples.
- Dataset generated by Matlab: {BPSK, QPSK, 8PSK, 2FSK, 4FSK, 8FSK, 16QAM, 128QAM, 256QAM}, the signal to noise ratio (SNR) ranging from -10 dB to 20 dB with 2 dB interval is used. The number of samples is 720,000. In addition, frequency, phase, and timing offset are introduced as $f_0 = 0.1$, $\theta = \pi/16$, and $\tau = 7$.

2) Simulation setting: We assume that there are 10 EDs and 1 CD. Hence, each dataset is divided into 10 equal parts to simulate 10 different EDs. The number of the training epochs is set as 500. The parameters $\{\alpha, \beta_1, \beta_2\}$ for Adam are preset values of Keras, i.e., $\{0.001, 0.9, 0.999\}$. We use learning rate decay to train the model, where the learning rate will decline by 80% after the 10 training epochs model is not improved. The experiment platform is GTX 2080Ti and the DL framework is Tensorflow 1.10.0 with Keras 2.2.4.

TABLE II
THE PARAMETERS, MODEL WEIGHT, P_{CC} OF MCMBNN ON RML2016.10A UNDER DIFFERENT U.

Metric \ U	32	64	128	256
Parameter	**48,934**	63,494	111,046	279,878
Model weight	**316 kB**	375 kB	565 kB	1.2 MB
$\mathbf{P_{cc}}$	57.08%	55.82%	**62.84%**	62.66%

TABLE III
THE PARAMETERS, MODEL WEIGHT, P_{CC} OF MCMBNN ON RML2016.10A UNDER DIFFERENT K_s^{group}.

Metric \ K_s^{group}	1×1	3×3	5×5	7×7
Parameter	**91,046**	111,046	151,046	211,046
Model weight	**485 kB**	565 kB	725 kB	965 kB
$\mathbf{P_{cc}}$	57.46%	**62.84%**	62.27%	62.77%

B. The Setting of Key Hyper-parameters

The units of GRU U and the size of group convolution kernel K_s^{group} are two key hyper-parameters, which influence the model complexity and classification performance of MCMBNN. As is shown in TABLE II, we set different U to make a balance between model complexity and classification performance. With the increasing the number of U, the classification performance of MCMBNN is improved obviously. Meanwhile, the model complexity of MCMBNN becomes higher. Therefore, we adopted a compromise between classification performance and model lightweight and set U as 128. Similarly, as is shown in TABLE III, it is appropriate to set K_s^{group} as 3×3.

C. Performance Comparison Between Different Networks with CentAMC

1) Classification performance: The correct classification probability (PCC) is generally adopted to describe the recognition and classification performance, which can be written as

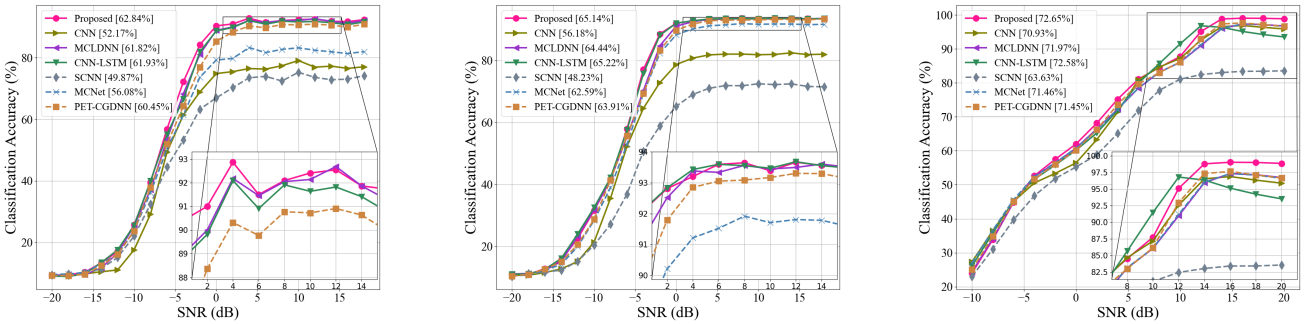
$$P_{cc}^i = \frac{N_{correct}^i}{N_{test}} \times 100\%, \tag{12}$$

where P_{cc}^i is the correct classification probability at SNR $= i$ dB, and $N_{correct}^i$ is the number of corrected classification samples at SNR $= i$ dB, and N_{test} is the number of the testing samples. The average value of P_{cc}^i is denoted as P_{cc}.

Fig. 9 shows the comparison of classification performance between the proposed MCMBNN and five SoA networks based on three datasets. Fig. 9(a) and Fig. 9(b) show that the proposed MCMBNN has an obvious advantage over the other networks when -6 dB $\leqslant$ SNR $\leqslant$ 0 dB. Fig. 9(c) shows that the proposed MCMBNN also has a best classification accuracy, which proves MCMBNN can also distinguish the signal with time-varying frequency offset. In addition, compared with MCNet and SCNN, the other networks perform better in the classification accuracy because they can make full use of the temporal features of the signals.

To analyze the classification accuracy of different modulation signals, classification performance of various modulation signals in different networks at 0 dB is given in TABLE IV. The MCMBNN performs better in recognizing BPSK, CPFSK and 64-QAM compared with other SoA Networks. For a further analysis, six confusion matrix evaluations with SNR $= 0$ dB based on different networks are presented in Fig. 10. The rows of the confusion matrix represent the real modulation types, while the columns represent the predicted. We can observe that 16-QAM and 64-QAM are confused, and similar confusion also occurs between WBFM and AM-DSB. The reason for the confusion between 16-QAM and 64-QAM is that as the order of QAM becomes higher, the constellation points also become denser. For WBFM and AM-DSB, both of them belong to continuous analog modulation signals and the features that can distinguish them are very weak. Although these confusion problems still exist in each network, it has been greatly improved in the networks that include temporal feature extraction units, especially for MCMBNN.

2) Model complexity: The complexity of the neural network is described by parameters and model weight in this paper. As is shown in TABLE V, compared with three high-performance networks, MCMBNN shows better classification performance with a lower model complexity. Compared with

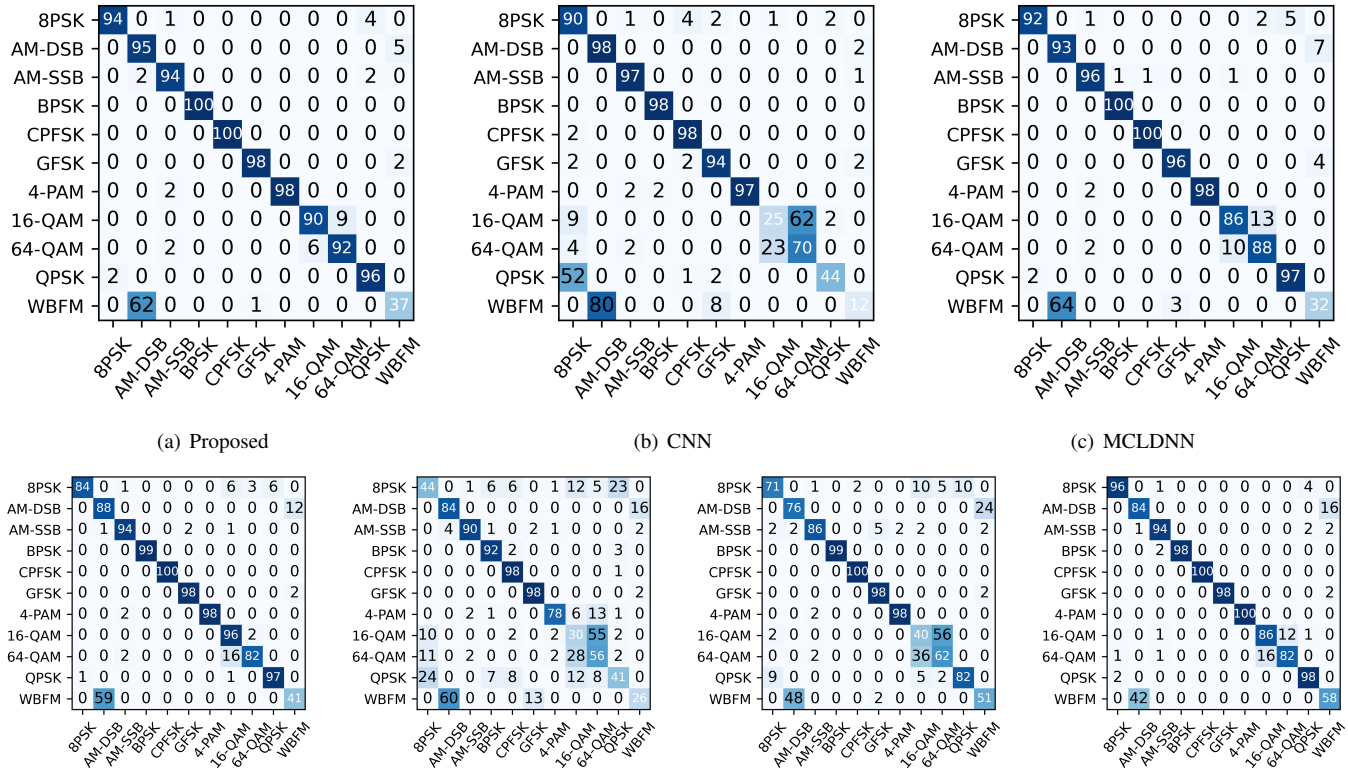


(a) (b) (c)

Fig. 9. Classification performance evaluation of the CentAMC method based on three typical datasets: (a) RadioML.2016.10A. (b) RadioML.2016.10B. (c) Dataset generated by Matlab.

TABLE IV
CLASSIFICATION PERFORMANCE OF DIFFERENT NETWORKS BASED ON RADIOML.2016.10A WHEN SNR = 0 DB BASED ON RML2016.10A.

	8PSK	AM-DSB	AM-SSB	BPSK	CPFSK	GFSK	PAM4	16-QAM	64-QAM	QPSK	WBFM
MCMBNN	93.5	95.0	94.0	**99.5**	**100.0**	97.5	98.5	90.0	**92.5**	96.5	37.0
MCMBNN-A	77.5	85.0	85.0	98.5	99.5	91.0	75.0	40.0	44.0	79.0	27.0
MCMBNN-B	93.0	96.0	**97.0**	**99.5**	**100.0**	98.0	98.5	84.0	78.5	94.5	35.5
MCMBNN-C	92.5	88.5	96.0	99.0	**100.0**	93.5	98.5	89.0	**92.5**	94.0	40.0
MCMBNN-D	88.0	90.5	93.5	99.0	**100.0**	**99.5**	98.5	87.5	85.5	98.5	39.0
MCMBNN-E	**96.5**	94.5	95.0	98.0	**100.0**	97.0	99.0	92.0	92.0	**99.5**	48.5
CNN	89.5	**98.0**	**97.0**	98.5	98.5	93.5	97.0	25.0	70.0	44.5	12.0
MCLDNN	91.5	93.0	95.5	**99.5**	**100.0**	96.5	98.5	86.0	88.0	97.0	32.5
CNNLSTM	84.0	87.5	93.5	99.0	99.5	98.0	98.5	**96.0**	82.0	97.0	41.0
SCNN	44.0	83.5	89.5	92.5	98.0	98.0	77.5	30.0	56.0	41.0	26.5
MCNET	71.0	75.5	85.5	99.0	99.5	97.5	98.5	40.5	61.5	82.5	51.0
PET-CGDNN	95.5	83.5	94.5	98.0	**100.0**	98.5	**99.5**	86.0	82.0	98.0	**57.9**



(a) Proposed (b) CNN (c) MCLDNN

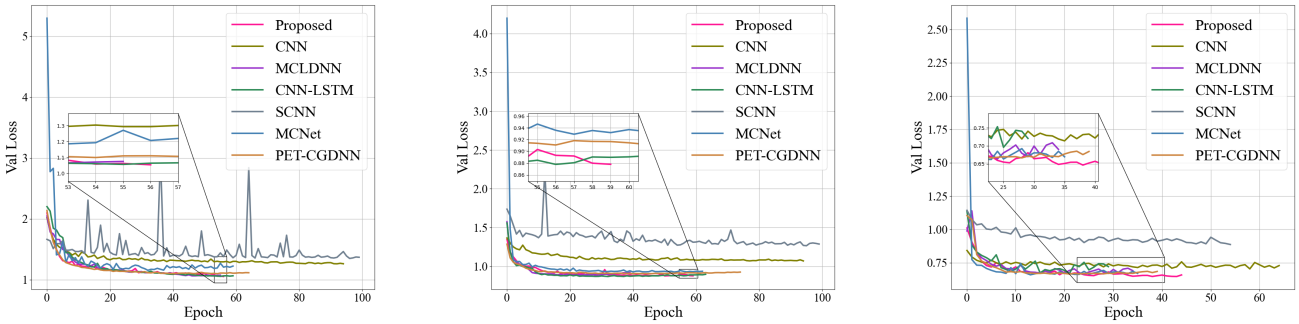
(d) CNNLSTM (e) SCNN (f) MCNET (g) PET-CGDNN

Fig. 10. Confusion matrix evaluation of different neural networks based on RadioML.2016.10A when SNR = 0 dB.

TABLE V
THE PARAMETERS, MODEL WEIGHT AND P_{CC} OF THE DIFFERENT NETWORKS BASED ON THREE TYPICAL DATASETS.

Neural Network	Parameters	Model weight	$\mathbf{Pcc_1}$/$\mathbf{Pcc_2}$/$\mathbf{Pcc_3}$
MCLDNN	406,199	1.669 MB	61.82% 64.44% 71.97%
CNN	2,190,283	8.793 MB	52.17% 56.18% 70.93%
CNN-LSTM	332,875	1.396 MB	61.93% **65.22%** 72.58%
SCNN	104,395	445 kB	49.87% 48.23% 63.63%
MCNet	90,763	463 kB	56.08% 62.59% 71.46%
PET-CGDNN	**71,871**	**324 kB**	60.45% 63.91% 71.45%
Proposed	111,046	565 kB	**62.84%** 65.14% **72.65%**

Note: Pcc_1, Pcc_2 and Pcc_3 are P_{CC} respectively based on RadioML.2016.10A, RadioML.2016.10B and Dataset generated by Matlab with CentAMC.



(a) (b) (c)

Fig. 11. Validation loss evaluation of of the CentAMC method based on three typical datasets: (a) RadioML.2016.10A. (b) RadioML.2016.10B. (c) Dataset generated by Matlab.

three lightweight networks, MCMBNN sacrifices a little model complexity in exchange for greater classification performance improvement, which is acceptable for the computing capability and memory of EDs (such as Raspberry Pi 3B used in [54]). In addition, the model weight of MCMBNN takes up less storage compared with two high-performance networks, which means that it is appropriate to deploy MCMBNN to EDs.

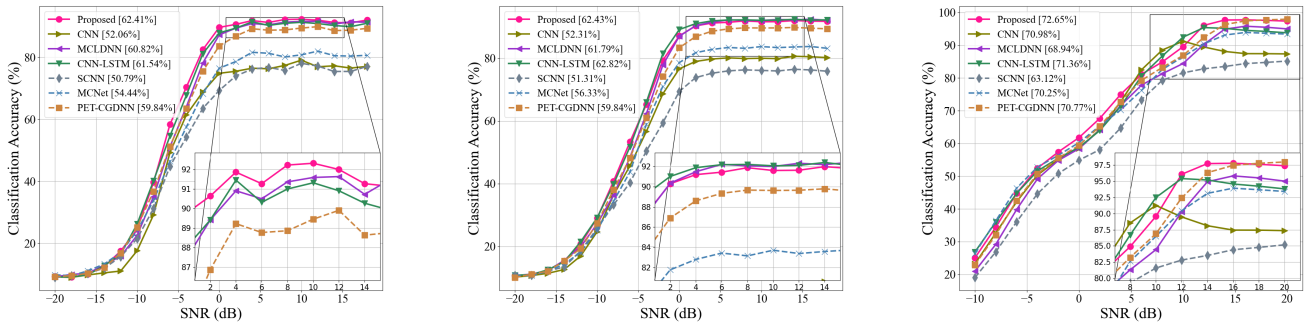
3) Loss evaluation: The validation loss within 100 epochs based on three datasets are given in Fig. 11. The validation loss of the MCMBNN keep a relatively stable convergence speed, which is consistent with that of other high-performance networks. It indicates that the network is perfectly trained and can adapt to different datasets. For further explanation, high-performance networks have a deeper or wider network structure compared with lightweight networks, which is helpful to fully extract features. However, high-performance networks are also easy to overfitting due to a large amount of parameters, which rarely occurs in lightweight networks.

TABLE VI
THE PARAMETERS, MODEL WEIGHT, P_{CC} OF ABLATION STUDY BASED ON RML2016.10A.

Neural Network	Parameters	Model weight	$\mathbf{P_{cc}}$
MCMBNN-A	181,318	843 kB	51.23%
MCMBNN-B	134,086	657 kB	62.33%
MCMBNN-C	133,596	647 kB	62.54%
MCMBNN-D	110,789	527 kB	62.12%
MCMBNN-E	**94,119**	**440 kB**	62.27%
Proposed	111,046	565 kB	**62.84%**

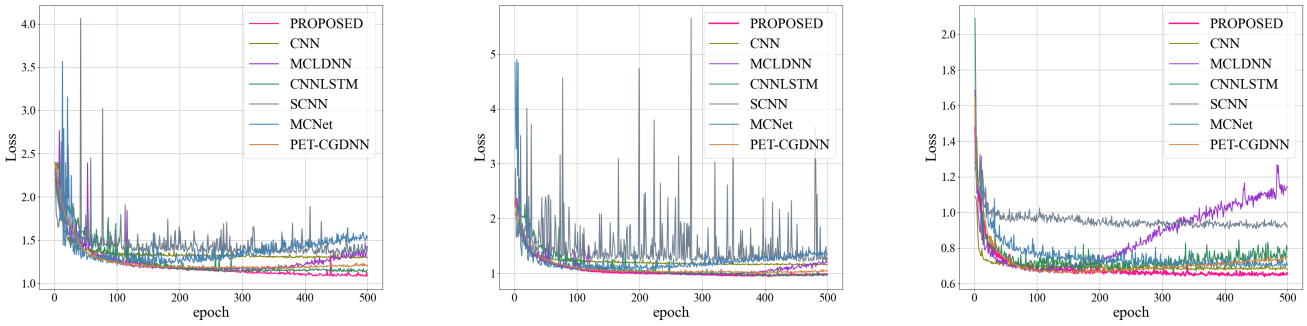
4) Ablation study: Ablation study is to observe how the different parts of the network affect the overall performance through deleting or replacing parts of the network. We conducted four groups of ablation experiments respectively, namely as: MCMBNN-A (GRU layer is replaced by one Flatten layer), MCMBNN-B (GRU layer is replaced by one LSTM layer), MCMBNN-C (group convolution layer is replaced by standard convolution layer), MCMBNN-D (PET is removed) and MCMBNN-E (multi-channel learning framework is replaced by inputting IQ signals directly).

As is shown in TABLE VI, the proposed MCMBNN has the best classification performance compared with other ablation experiments, which means that the advantages of different parts of MCMBNN are complementary and their combination results in superior performance. Specifically, MCMBNN and MCMBNN-B outperform MCMBNN-A in classification performance, which means that the temporal feature extraction layers (GRU layer or LSTM layer) can extract rich temporal features for modulation recognition. Moreover, it proves to be meaningful to design lightweight networks with one GRU layer instead of LSTM layer from the insight of reducing the model complexity by comparing MCMBNN with MCMBNN-B. Through the comparison of MCMBNN and MCMBNN-C, we can observe that group convolution not only reduces the complexity of network, but also has a similar classification performance with standard convolution, which proves that it is feasible to design lightweight networks with group convolution. Next, MCMBNN-D has a disadvantage in classification performance compared with MCMBNN. In particular, MCMBNN-D is inferior in recognizing 8PSK,



(a) (b) (c)

Fig. 12. Classification performance evaluation of the DecentAMC method using three typical datasets: (a) RadioML.2016.10A. (b) RadioML.2016.10B. (c) Dataset generated by Matlab.



(a) (b) (c)

Fig. 13. Validation loss evaluation of the DecentAMC method using three typical datasets: (a) RadioML.2016.10A. (b) RadioML.2016.10B. (c) Dataset generated by Matlab.

AM-DSB and 64-QAM shown in TABLE IV compared with MCMBNN, which indicates that PET is working for extracting phase offset information for a better classification performance. Finally, multi-channel learning framework is adopted in MCMBNN, which is helpful to extract the complementary features compared with MCMBNN-E. Hence, we can conclude that GRU layer, PET and multi-channel learning framework play a significant role in the performance improvement and group convolution can be adopted to decrease the model complexity.

D. Performance Comparison between Different Networks with DecentAMC

1) Classification performance: Here, the classification performance of DecentAMC based on the different Networks are compared, which is shown in Fig .12. Similar to CentAMC, MCMBNN also shows superior classification performance across the three datasets, which proves the high robustness of MCMBNN. In addition, there is a maximum performance gap of 4% between DecentAMC and CentAMC because model weight is shared rather than dataset for avoiding data privacy leaking from EDs and reducing the computing pressure of CD in DecentAMC.

2) Loss evaluation: Fig .13 presents the validation loss of the DecentAMC, where we can observe that different networks have a similar convergence speed, and hence the convergence global epoch is similar. The communication overhead of DecentAMC is defined as

$$C_{Decent} = 2NW_mT, \tag{13}$$

where N demotes the number of EDs, W_m represents the size of global model weight, and T represents the number of convergence global epoch. It is easy to prove that there exists a positive correlation between communication overhead and model weight when the number of convergence global epochs is the same. This is the reason why deploying lightweight network to EDs with DecentAMC method can decrease communication overhead. Besides, the loss curve of MCMBNN is more stable and keeps a downward trend until convergence compared with other networks, which indicates that MCMBNN has a good fitting performance (MCLDNN has not decreased but increased because overfitting occurs).

VI. Conclusions

In this paper, we proposed a lightweight neural network named MCMBNN and a decentralized training method named the DcentAMC method. The MCMBNN is designed through a combination of three lightweight blocks (PET block, spatial information extraction block and temporal feature extraction & Softmax block), which respectively adopt different lightweight

ideas for further achieving a overall lightweight network. The DecentAMC method adopts FL to train the model for AMC in order to acieve decentralized learning. Experimental results show that the MCMBNN can achieve a superior classification performance with a low model complexity based on the advantages of multiple blocks. In addition, the MCMBNN based DcentAMC method can be deployed on EDs with a relatively better classification performance and a lower communication overhead. This will address the problem of EDs' insufficient computing power and data leakage associated with traditional DL model and CentAMC method. Future work will attempt to quantify the model weight to further reduce the communication overhead. We also conduct several experiments based on datasets that is more representative of realistic scenario.

REFERENCES

[1] M. Stoyanova, Y. Nikoloudakis, S. Panagiotakis, E. Pallis and E. K. Markakis, "A survey on the internet of things (IoT) forensics: challenges, approaches, and open issues," *IEEE Commun. Surv. Tut.*, vol. 22, no. 2, pp. 1191–1221, Secondquarter 2020.

[2] L. Liu, Y. Zhou, J. Yuan, W. Zhuang, and Y. Wang, "Economically Optimal MS association for multimedia content delivery in cache-enabled heterogeneous cloud radio access networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 7, pp. 1584–1593, Jul. 2019.

[3] L. Liu, Y. Zhou, W. Zhuang, J. Yuan and L. Tian, "Tractable coverage analysis for hexagonal macrocell-based heterogeneous UDNs with adaptive interference aware CoMP," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 503–517, Jan. 2019.

[4] M. Rasori, M. La Manna, P. Perazzo and G. Dini, "A survey on attribute-based encryption schemes suitable for the internet of things," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8269–8290, June 2022.

[5] R. Zhao, Y. Wang, *et al*., "Semi-supervised federated learning based intrusion detection method for internet of things," *IEEE Internet Things J.*, early access, doi: 10.1109/JIOT.2022.3175918

[6] S. I. Popoola, B. Adebisi, *et al*., "Hybrid deep learning for botnet attack detection in the internet of things networks," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4944–4956, Jun. 2021.

[7] R. Zhao, *et al*., "A novel intrusion detection method based on lightweight neural network for internet of things," *IEEE Internet Things J.*, early access, doi: 10.1109/JIOT.2021.3119055

[8] Z. Qin, X. Zhou, L. Zhang, Y. Gao, Y. Liang, and G.-Y. Li, "Survey of automatic modulation classification techniques: classical approaches and new trends," *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 1, pp. 6–20, Mar. 2020.

[9] Q. Wang, H. Sun, R. Q. Hu and A. Bhuyan, "When machine learning meets spectrum sharing security: methodologies and challenges," *IEEE Open j. Commun. Soc.*, vol. 3, pp. 176–208, 2022.

[10] C. S. Long, K. M. Chugg, and A. Polydoros, "Further results in likelihood classification of QAM signals," in *Proc. Military Commun. Conf.*, Fort Monmouth, NJ, USA, 1994, pp. 57–61.

[11] F. Hameed, O. A. Dobre, and D. C. Popescu, "On the likelihood-based approach to modulation classification," *IEEE Trans. Wireless Commun.*, vol. 8, no. 12, pp. 5884–5892, Dec. 2009.

[12] A. Polydoros and K. Kim, "On the detection and classification ofquadrature digital modulations in broad-band noise," *IEEE Trans. Commun.*, vol. 38, no. 8, pp. 1199–1211, Aug. 1990.

[13] P. Panagiotou, A. Anastasopoulos, and A. Polydoros, "Likelihood ratio tests for modulation classification," in *Proc. IEEE Mil. Commun. Conf.*, Oct. 2000, vol. 2, pp. 670–674.

[14] A. O. Abdul Salam, R. E. Sheriff, S. R. Al-Araji, K. Mezher and Q. Nasir, "A unified practical approach to modulation classification in cognitive radio using likelihood-based techniques," in *2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2015, pp. 1024–1029.

[15] J. Zheng and Y. Lv, "Likelihood-based automatic modulation classification in OFDM with index modulation," *IEEE Trans. Veh. Technol.*,vol. 67, no. 9, pp. 8192–8204, Sep. 2018.

[16] N. Jafar, A. Paeiz and A. Farzaneh, "Automatic modulation classification using modulation fingerprint extraction," *J. Syst. Electron.*, vol. 32, no. 4, pp. 799–810, Aug. 2021.

[17] R. Gupta, S. Majhi and O. A. Dobre, "Design and implementation of a tree-based blind modulation classification algorithm for multiple-antenna systems," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 8, pp. 3020–3031, Aug. 2019.

[18] M. Oner and O. A. Dobre, "On the second-order cyclic statistics of signals in the presence of receiver impairments," *IEEE Trans. Commun.*, vol. 59, no. 12, pp. 3278–3284, Dec. 2011.

[19] Wu, Hsiao-Chun, Mohammad Saquib, and Zhifeng Yun. "Novel automatic modulation classification using cumulant features for communications via multipath channels." *IEEE Trans. Wirel. Commun.*, vol. 7, no. 8, pp. 3098–3105, August 2008.

[20] K. C. Ho, W. Prokopiw, and Y. T. Chan, "Modulation identification of digital signals by the wavelet transform," *IEE Proc. Radar Sonar Navig.*, vol. 147, no. 4, pp. 169–176, Aug. 2000.

[21] A. Swami and B. M. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Trans. Commun.*, vol. 48, no. 3,pp. 416–429, Mar. 2000.

[22] M. W. Aslam, Z. Zhu and A. K. Nandi, "Automatic modulation classification using combination of genetic programming and KNN," *IEEE Trans. Wirel. Commun.*, vol. 11, no. 8, pp. 2742–2750, Aug. 2012.

[23] B. Sun, Y. Zhou, J. Yuan, Y. -F. Liu, L. Wang and L. Liu, "High order PSK modulation in massive MIMO systems with 1-bit ADCs," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2652–2669, Apr. 2020.

[24] Q. Hu, F. Gao, H. Zhang, S. Jin, and G.-Y. Li, "Deep learning for channel estimation: interpretation, performance, and comparison," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2398–2412, Apr. 2021.

[25] Y. Zhou, L. Liu, L. Wang, N. Hui, X. Cui, J. Wu, Y. Peng, Y. Qi, C. Xing, "Service-aware 6G: An intelligent and open network based on the convergence of communication, computing and caching," *Digital Commun. Netw.*, vol 6, no. 3, pp. 253–260, Sept. 2020.

[26] Y. Su, Y.Q. Liu, Y. Zhou, J. Yuan and J. L. Shi, "Broadband LEO satellite communications: architectures and key technologies," *IEEE Wireless Commun.*, vol. 26, no. 2, pp. 55–61, Apr. 2019.

[27] M. Abdel-Basset, H. Hawash, R. K. Chakrabortty and M. J. Ryan, "Semi-supervised spatiotemporal deep learning for intrusions detection in IoT networks," *IEEE Internet Things J.*, vol. 8, no. 15, pp. 12251–12265, Aug. 2021.

[28] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa and F. T. H. d. Hartog, "ToN_IoT: the role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion data sets," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 485–496, Jan. 2022.

[29] B. Yuan, J. Wang, P. Wu and X. Qing, "IoT malware classification based on lightweight convolutional neural networks," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3770–3783, Mar. 2022.

[30] F. Meng, P. Chen, L. Wu and X. Wang, "Automatic modulation classification: a deep learning enabled approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10760–10772, Nov. 2018.

[31] S. Lin, Y. Zeng and Y. Gong, "Learning of time-frequency attention mechanism for automatic modulation recognition," *IEEE Wireless Commun. Lett.*, vol. 11, no. 4, pp. 707–711, Apr. 2022.

[32] S. Chang, S. Huang, R. Zhang, Z. Feng, and L. Liu, "Multi-task learning based deep neural network for automatic modulation classification," *IEEE Internet Things J.*, early access, doi: 10.1109/JIOT.2021.3091523

[33] Y. Wang, L. Guo, Y. Zhao, J. Yang, B. Adebisi, H. Gacanin, and G. Gui, "Distributed learning for automatic modulation classification in edge devices," *IEEE Wireless Commun. Lett.*, vol. 9, no. 12, pp. 2177–2181, Dec. 2020.

[34] Z. Zhang, C. Wang, C. Gan, S. Sun and M. Wang, "Automatic modulation classification using convolutional neural network with features fusion of SPWVD and BJD," *IEEE Trans. Signal Process.*, vol. 5, no. 3, pp. 469–478, Sept. 2019.

[35] S. Hanna, C. Dick and D. Cabric, "Signal processing-based deep learning for blind symbol decoding and modulation classification," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 82–96, Jan. 2022.

[36] S.-H. Kim, C.-B. Moon, J. -W. Kim and D. -S. Kim, "A hybrid deep learning model for automatic modulation classification," *IEEE Wireless Commun. Lett.*, vol. 11, no. 2, pp. 313–317, Feb. 2022.

[37] Y. Wang, G. Gui, T. Ohtsuki, and F. Adachi, "Multi-task learning for generalized automatic modulation classification under non-Gaussian noise with varying SNR conditions," *IEEE Trans. Wireless Commu.*, vol. 20, no. 6, pp. 3587–3596, Jun. 2021.

[38] X. Liu, Q. Wang and H. Wang, "A Two-Fold group lasso based lightweight deep neural network for automatic modulation classification," in *IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6.

[39] X. Zhang, *et al*., "NAS-AMR: neural architecture search based automatic modulation recognition for integrated sensing and communication systems," *IEEE Trans. Cogn. Commun. Netw.*, early access, doi: 10.1109/TCCN.2022.3169740.
[40] P. Qi, X. Zhou, S. Zheng, and Z. Li, "Automatic modulation classification based on deep residual networks with multimodal information," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 1, pp. 21–33, Mar. 2021.
[41] Y. Wang, J. Yang, M. Liu, and G. Gui, "LightAMC: lightweight automatic modulation classification via deep learning and compressive sensing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3491–3495, Mar. 2020.
[42] Y. Lin, Y. Tu, and Z. Dou, "An improved neural network pruning technology for automatic modulation classification in edge device," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5703–5706, May. 2020.
[43] T. Huynh-The, C. Hua, Q. Pham and D. Kim, "MCNet: an efficient CNN architecture for robust automatic modulation classification," *IEEE Commun. Lett.*, vol. 24, no. 4, pp. 811–815, April 2020.
[44] Z. Zhang, H. Luo, C. Wang, C. Gan and Y. Xiang, "Automatic modulation classification using CNN-LSTM based dual-stream structure," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13521–13531, Nov. 2020.
[45] J. Xu, C. Luo, G. Parr and Y. Luo, "A spatiotemporal multi-channel learning framework for automatic modulation recognition" *IEEE Wireless Commun. Lett.*, vol. 9, no. 10, pp. 1629–1632, Oct. 2020.
[46] F. Zhang, C. Luo, J. Xu and Y. Luo, "An efficient deep learning model for automatic modulation recognition based on parameter estimation and transformation," *IEEE Commun. Lett.*, vol. 25, no. 10, pp. 3287–3290, Oct. 2021.
[47] X. Fu, *et al*., "Lightweight automatic modulation classification based on decentralized learning," *IEEE Trans. Cogn. Commun. Netw.*, vol. 8, no. 1, pp. 57–70, Mar. 2022.
[48] T. J. Oshea and N. West, "Radio machine learning dataset generation with GNU radio," in *Proc. GNU Radio Conf*., 2016, vol. 1, no. 1, pp. 1–6.
[49] C. F. Teng, C. Y. Chou, C. H. Chen and A. Y. Wu, "Accumulated polar feature-based deep learning for efficient and lightweight automatic modulation classification with channel compensation mechanism," *IEEE Trans. Veh. Technol*, vol. 69, no. 12, pp. 15472–15485, Dec. 2020.
[50] S. Luan, Y. Gao, J. Zhou and Z. Zhang, "Automatic modulation classification based on cauchy-score constellation and lightweight network under impulsive noise," *IEEE Wireless Commun. Lett.*, vol. 10, no. 11, pp. 2509–2513, Nov. 2021.
[51] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Nevada, USA, Dec. 3-6, 2012, pp. 1097–1105.
[52] H. Wei, Z. Wang and G. Hua, "Dynamically mixed group convolution to lighten convolution operation," *ICAIBD*, vol. 25, pp. 203–206, 2021.
[53] He, Kaiming, and Jian Sun. "Convolutional neural networks at constrained time cost," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2818–2826.
[54] E. S. Lage, R. L. Santos, S. M. T. Junior and F. Andreotti, "Low-cost IoT surveillance system using hardware-acceleration and convolutional neural networks," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 931–936.
[55] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, Cadiz, Spain, May 9-11, 2016, pp. 1–11.
[56] T. L. Burns, *et al*., "Evaluating deep learning networks for modulation recognition," in *IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, 2021, pp. 25–32.

Biao Dong (Graduate Student Member, IEEE) received the B.S. degree from Jiangsu University of Technology, Changzhou, China, in 2020. He is currently a postgraduate student at Nanjing University of Posts and Telecommunications, Nanjing, China. His research interests include statistical signal processing, machine learning, deep learning, distributed/decentralized learning.

Yuchao Liu received the B.S. degree from Central South University, Changsha, China, in 2007 and received the M.S. degree from China Research Institute of Radiowave Propagation, Qingdao, China, in 2010. After that, he joined the Institute as for Engineer as well as Senior Engineer. His research interests include deep learning for wireless communications.

Guan Gui (Senior Member, IEEE) received the Ph.D. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 2012. From 2009 to 2014, he joined the Tohoku University as a research assistant as well as a postdoctoral research fellow, respectively. From 2014 to 2015, he was an Assistant Professor in the Akita Prefectural University. Since 2015, he has been a professor with Nanjing University of Posts and Telecommunications, Nanjing, China. His recent research interests include intelligent sensing and identification.

Xue Fu (Graduate Student Member, IEEE) received the B.S. degree from China Agricultural University, Beijing, China, in 2020. She is currently a postgraduate student at Nanjing University of Posts and Telecommunications, Nanjing, China. Her research interests include statistical signal processing, machine learning, deep learning, distributed/decentralized learning.

Heng Dong (Member, IEEE) received the B.S. and M.S. degrees in electronics engineering from Southeast University, Nanjing, China, in 1992 and 1995, respectively. Since 1996, he has been with the Department of Telecommunications Engineering, Nanjing University of Posts and Telecommunications, where he is currently an Associate Professor. His research interests include wireless communications theory, digital signal processing, and Internet-of-things applications.

Bamidele Adebisi (Senior Member, IEEE) received his Bachelor's degree in electrical engineering from Ahmadu Bello University Zaria, Nigeria, in 1999, and his Master's degree in advanced mobile communication engineering and Ph.D. degree in communication systems from Lancaster University, United Kingdom, in 2003 and 2009, respectively. He was a senior research associate with the School of Computing and Communication, Lancaster University, from 2005 to 2012. He joined Manchester Metropolitan University in 2012, where he is currently a Full Professor (Chair) of intelligent infrastructure systems.

Haris Gacanin (Fellow, IEEE) received his Dipl.-Ing. degree in Electrical engineering from the University of Sarajevo in 2000. In 2005 and 2008, respectively, he received MSc and Ph.D. from Tohoku University in Japan. He was with Tohoku University from 2008 until 2010 first as Japan Society for the Promotion of Science (JSPS) postdoctoral fellow and later, as an Assistant Professor. He joined Alcatel-Lucent Bell (now Nokia Bell) in 2010 as a Physical-layer Expert and later moved to Nokia Bell Labs as Department Head. Since April 2020, he joined RWTH Aachen University. He is a head of the Chair for Distributed Signal Processing and co-director of the Institute for Communication Technologies and Embedded Systems.

Hikmet Sari (Life Fellow, IEEE) received the engineering diploma and the Ph.D. degree from ENST, Paris, France, and the Habilitation degree from the University of Paris XI. From 1980 to 2002, he held various research and management positions at Philips Research Laboratories, SAT, Alcatel, Pacific Broadband Communications, and Juniper Networks. From 2003 to 2016, he was a Professor and the Head of the Telecommunications Department, Suplec, and a Chief Scientist at Sequans Communications. He is currently a Professor with the Nanjing University of Posts and Telecommunications (NJUPT).