Automatic Modulation Classification: A Deep Learning Enabled Approach

Fan Meng, Peng Chen, *Member, IEEE*, Lenan Wu, and Xianbin Wang, *Fellow, IEEE*

***Abstract*—Automatic modulation classification (AMC), which plays critical roles in both civilian and military applications, is investigated in this paper through a deep learning approach. Conventional AMCs can be categorized into maximum likelihood (ML) based (ML-AMC) and feature-based AMC. However, the practical deployment of ML-AMCs is difficult due to its high computational complexity, and the manually extracted features require expert knowledge. Therefore, an end-to-end convolution neural network (CNN) based AMC (CNN-AMC) is proposed, which automatically extracts features from the long symbol-rate observation sequence along with the estimated signal-to-noise ratio (SNR). With CNN-AMC, a unit classifier is adopted to accommodate the varying input dimensions. The direct training of CNN-AMC is challenging with the complicated model and complex tasks, so a novel two-step training is proposed, and the transfer learning is also introduced to improve the efficiency of retraining. Different digital modulation schemes have been considered in distinct scenarios, and the simulation results show that the CNN-AMC can outperform the feature-based method, and obtain a closer approximation to the optimal ML-AMC. Besides, CNN-AMCs have the certain robustness to estimation error on carrier phase offset and SNR. With parallel computation, the deep-learning-based approach is about 40 to 1700 times faster than the ML-AMC regarding inference speed.**

***Index Terms*—Automatic modulation classification, convolution neural network, deep learning, two-step training.**

I. Introduction

IN ACHIEVING the maximum possible transmission rates, adaptive modulation schemes are used in the civilian wireless communication system for full utilization of time-varying channels. Under these circumstances, information concerning a specific modulation scheme used in a communication process has to be shared by the transmitter with the receiver through a network protocol, at the cost of protocol overhead. Such overhead could be eliminated when the receiver has the capability of modulation scheme recognition. On the other hand, many military applications also require automatic discovery of the modulation schemes used by signals from adversaries. Such applications encompass signal interception and jamming. As a result, automatic modulation classification (AMC) [1], [2] is of great importance for both civilian and military applications in achieving automatic receiver configuration, interference mitigation, and spectrum management.

The traditional feature-based AMC algorithms [3], [4] can be realized by three steps: data preprocessing, feature extraction and classificatory decision. Since good statistical features can provide robust performance with low complexity [5], feature extraction-based AMC methods have been studied in many existing papers [6], [7], such as cyclic feature [8], higher order moments and cumulants [9]–[11], wavelet transforms [12], [13], *et al*. In classificatory decision, the conventional classifiers includes random forest [14], multi-layer perception [15], [16] and support vector machine (SVM) [17], [18]. These AMCs are combinations of different feature extraction algorithms and classifications, and require both expert knowledge and artificial design.

The AMC methods based on maximum likelihood (ML) are also studied in [19], where the ML-AMCs have been proved to achieve the optimal performance in the scenarios with mathematical channel models. Different from the feature-based AMCs, ML-AMCs calculate likelihood functions of all candidate modulations, and choose the modulation scheme with maximal likelihood value, working as an end-to-end module. However, the high computational complexity makes it extremely difficult for the real-time implementation with low cost [20]. Besides, the exact likelihoods in ML-AMC are hard to achieve in the complex environments.

In feature-based AMCs, the machine learning methods simply work as a mapping function between features and multi-hypothesis. To obviate feature engineering, the deep learning method [21] is developing rapidly in recent years both in algorithm design and hardware implementation [22], and it can learn significantly more complex functions than a shallow one. The rapid development of deep learning has led to some successful applications in communications [23]–[25], including modulation classification. In [26], the eye diagram of the raw signal is used as input for Lenet-5-based classifier [27] and subtly re-

Manuscript received January 15, 2018; revised May 13, 2018 and July 12, 2018; accepted August 12, 2018. Date of publication September 4, 2018; date of current version November 12, 2018. This work was supported in part by the National Natural Science Foundation of China under Grants 61801112, 61271204, 61471117, and 61601281, in part by the Natural Science Foundation of Jiangsu Province under Grant SBK2018042259, and in part by the open program of the State Key Laboratory of Millimeter Waves in Southeast University under Grant Z201804. The review of this paper was coordinated by Prof. Guan Gui. *(Corresponding author: Peng Chen.)*

F. Meng and L. Wu are with the School of Information Science and Engineering, Southeast University, Nanjing 210096, China (e-mail: mengxiaomaomao@outlook.com; wuln@seu.edu.cn).

P. Chen is with the State Key Laboratory of Millimeter Waves, Southeast University, Nanjing 210096, China (e-mail: chenpengseu@seu.edu.cn).

X. Wang is with the Department of Electrical and Computer Engineering, Western University, London, ON N6A 3K7, Canada (e-mail: xianbin.wang@uwo.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2018.2868698

0018-9545 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

lates the problem of AMC with the well-studied area of image recognition. The convolutional network for radio modulation recognition is proposed in [28], [29], and the simulations with a dataset called RML2016.10b show that the classification accuracy is higher than those with expert feature engineering. Then in [30], a LSTM-based AMC is proved to outperform the CNN model with oversampled received signals at small or medium scales. The long symbol-rate signal is studied in [31], and the stacked auto-encoders achieves excellent results with increased simulation runs.

In our paper, we propose a deep neural network (DNN) enabled AMC which can automatically learn to extract features from long symbol-rate signals at low SNR, and adopt the convolution neural network (CNN) to realize this AMC. The CNN-AMC can approximate the ML-AMC with minimal performance loss but significant speed improvement. The ML-AMC is used as a benchmark for our proposed CNN-AMC, and can also generate the training data in different scenarios. The contributions of this work are summarized as follows:

- Directly using the long symbol-rate signals at low SNR, we propose an end-to-end trainable AMC based on deep CNN. Different from the featured-based AMCs, the CNN-AMCs automatically learn to acquire features from the raw signals and simplify the AMC design. Besides, CNN-AMC makes an approximation to the ML-AMC, and the related processing can be accomplished parallelly to accelerate the inference speed.
- We introduce a two-step training method including pre-training and fine-tuning for the proposed CNN-AMC to solve the challenging direct training problem. For varying combinations of different modulation schemes and communication channel scenarios, the transfer learning is also proposed to improve the retraining efficiency further.
- By dividing long preprocessed observation sequence into signal segments with unit size, the unit classifier is introduced with the unit input dimension to accommodate the varying input dimensions, and parallelly realize the block calculation.

The remainder of this paper is organized as follows. Section II outlines the modulation classification problem in the framework of classical decision theory. In Section III the structure of a CNN-AMC featured with multiple inputs is provided, along with its corresponding two-step training, transfer learning and the unit classifier. Then, in Section IV, the proposed CNN-AMC, the ML-AMC and feature-based method are tested versus SNR in distinct environments, and the simulation results are analyzed. Conclusions and discussion are given in Section V.

II. System Model

A. Problem Formulation

Without loss of generality, the complex envelope of received baseband signal can be expressed as

$$y(t) = x(t; \mathbf{u}_k) + v(t), \tag{1}$$

where $v(t)$ is the complex additive white Gaussian noise (AWGN) with zero mean and double-sided power spectral density being $N_0/2$. The noiseless signal $x(t; \mathbf{u}_k)$ is modeled via a comprehensive form as

$$x(t; \mathbf{u}_k) = a e^{j(2\pi \Delta f t + \theta_c)} \sum_{n=0}^{N-1} x_n^{k,i} g(t - nT_s - \varepsilon T_s), \tag{2}$$

where $\mathbf{u}_k$ is defined as a multi-dimensional parameter set, in which a bunch of unknown signal and channel variables are stochastic or deterministic under the k-th modulation scheme, and it is given as

$$\mathbf{u}_k = \left\{ a, \theta_c, \Delta f, \{x^{k,i}\}_{i=1}^{M_k}, h(t), \varepsilon \right\}. \tag{3}$$

The symbols used in (2) and (3) are listed as follows:

1) a, the signal amplitude.
2) θ_c, the fixed phase offset introduced by the propagation delay as well as the initial carrier phase.
3) N, the number of symbols in received sequence, i.e., the observation sequence space.
4) $x^{k,i}$, the i-th constellation point under k-th modulation scheme where the symbols are equally distributed, $i \in \{1, \ldots, M_k\}, k \in \{1, \ldots, C\}$. The mark M_k denoted the symbol number using this modulation, and C is the number of candidate equiprobable modulation schemes, respectively. The modulated symbols are usually normalized with unit power

$$\frac{1}{M_k} \sum_{i=1}^{M_k} |x^{k,i}|^2 = 1. \tag{4}$$

   In addition, symbols in a data block are assumed to be independent and identical distributed (I.I.D).
5) Δf, the residual carrier frequency after carrier removal.
6) T_s, symbol interval.
7) $g(t)$, the composite effect of the residual channel $h(t)$ and the pulse-shaping function $p(t)$. It is expressed by the equation $g(t) = h(t) * p(t)$, where $*$ is the convolution operator. Usually, $p(t)$ is a root raised cosine pulse function with the duration T_s.
8) ε, the normalized epoch for timing offset between the transmitter and the receiver, $0 \le \varepsilon < 1$.

In this paper, the received signal is expressed as the sampled complex baseband representation, and SNR of E_s/N_0 is considered. The discrete observation sequence sampled at symbol interval T_s can be written as $\mathbf{y} = [y_0, \ldots, y_{N-1}]^T$, where the superscript T is the transpose operator. Each sampling instant of the received signal is the output of a pulse-shaping matched filter at the receiver, which can be expressed as

$$y_n = \frac{1}{T_s} \int_{nT_s}^{(n+1)T_s} y(t) p(t - nT_s)\, dt. \tag{5}$$

The pulse shape $p(t)$ of the transmitting signal is unknown for the receiver in the blind classification case. Similarly, instant y_n is usually normalized to unit power

$$y_n = \frac{y_n}{\sqrt{\frac{1}{N}\mathbf{y}^H \mathbf{y}}} \tag{6}$$

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

where the superscript H is the conjugate transpose operator. In our classifiers, the estimation of symbol SNR is required, which is equivalent to having knowledge of both a and N_0.

The conditional probability density function (PDF) of y_n under hypothesis H_k can be written as

$$p(y_n|H_k,\mathbf{u}_k)=\sum_{i=1}^{M_k}\frac{1}{M_k\sqrt{\pi N_0}}\exp\left(\frac{|y_n-x_n^{k,i}(\mathbf{u}_k)|^2}{N_0}\right)$$
$$\propto\frac{1}{M_k}\sum_{i=1}^{M_k}\exp\left(\frac{|x_n^{k,i}|^2-2\mathfrak{Re}\{y_n^*x_n^{k,i}(\mathbf{u}_k)\}}{N_0}\right). \tag{7}$$

The elements in parameter set $\mathbf{u}_k$ are deterministic or random variables with known PDFs. Hence, the marginalized density function of an observation sequence can be obtained by taking the statistical averaging of (7), and it is given as

$$\Gamma(y_n|H_k)=\mathbb{E}_{\mathbf{u}_k}\left\{\Gamma(y_n|H_k,\mathbf{u}_k)\right\}, \tag{8}$$

where $\mathbb{E}\{\cdot\}$ is the expectation operator.

The channel environment is assumed to vary slowly, and the parameters in set $\mathbf{u}_k$ is static over the whole observation period. Besides, the normalized epoch $\varepsilon=0$ and the noise is white, and thus the elements of $\mathbf{y}$ are I.I.D. The function $p(y_n|H_k,\mathbf{u}_k)$ represents the PDF of single observation at instant n, and the joint likelihood function of whole observation sequence can be described as

$$\Gamma(\mathbf{y}|H_k)=\prod_{n=0}^{N-1}p(y_n|H_k). \tag{9}$$

Floating point overflow in the numerical calculation can be introduced by a large N, and therefore the equivalent logarithmic form of (9) is usually adopted, and it is given by

$$\mathrm{L}(\mathbf{y}|H_k)=\ln\Gamma(\mathbf{y}|H_k). \tag{10}$$

According to the maximum likelihood criterion, given the observation sequence $\mathbf{y}$, the most possible hypothesis H_k which is correspondent to the k-th modulation scheme, is finally chosen with the maximal likelihood value

$$\widehat{k}=\arg\max_{k\in\{1,\ldots,C\}}\Gamma(\mathbf{y}|H_k), \tag{11}$$

where the top-mark $\hat{(\cdot)}$ denotes the estimation.

B. Correct Probability

The conditional correct probability $P_{\mathrm{cc}}(\mathbf{y}|H_k)$ is defined as

$$P_{\mathrm{cc}}(\mathbf{y}|H_k)=\frac{\Gamma(\mathbf{y}|H_k)}{\sum_{k'=1}^{C}\Gamma(\mathbf{y}|H_{k'})}. \tag{12}$$

Therefore, the overall correct probability P_{c} is represented as

$$P_{\mathrm{c}}=\frac{1}{C}\sum_{k=1}^{C}P_{\mathrm{cc}}(\mathbf{y}|H_k). \tag{13}$$

The distribution of observation sequence $\mathbf{y}$ is $f_{\mathbf{Y}}(\mathbf{y})$, then combine (12) and (13), the expectation of P_{c} is given by

$$\mathbb{E}_{\mathbf{Y}}\{P_{\mathrm{c}}\}=\int_{\mathbf{y}}P_{\mathrm{c}}f_{\mathbf{Y}}(\mathbf{y})\,d\mathbf{y}$$
$$=\frac{1}{C}\sum_{k=1}^{C}\int_{\mathbf{y}}\frac{\Gamma(\mathbf{y}|H_k)}{\sum_{k'=1}^{C}\Gamma(\mathbf{y}|H_{k'})}\Gamma(\mathbf{y}|H_k)\,d\mathbf{y}. \tag{14}$$

The parameter set $\mathbf{u}_k$ is assumed to be known, but in fact, it is usually modeled or estimated with some deviations. Then the (14) is given as

$$\mathbb{E}_{\mathbf{Y}}\{P_{\mathrm{c}}\}=\mathbb{E}_{\mathbf{U}_k,\hat{\mathbf{U}}_k,\mathbf{Y}}\left\{\frac{1}{C}\sum_{k=1}^{C}\int_{\mathbf{y}}\frac{\Gamma(\mathbf{y}|H_k,\hat{\mathbf{u}}_k)}{\sum_{k'=1}^{C}\Gamma(\mathbf{y}|H_{k'},\hat{\mathbf{u}}_{k'})}\right\}, \tag{15}$$

where $\hat{\mathbf{u}}_k$ is the estimation of $\mathbf{u}_k$. It is hard to obtain a closed-form solution of (14) and (15), but a numerical approximation can be obtained by Monte Carlo method instead.

C. Signal Segment

The normalized likelihood vector is defined as $\boldsymbol{\xi}^N=[\xi_1^N,\ldots,\xi_C^N]^T$, in which the element of corresponding modulation is $\xi_k^N=P_{\mathrm{cc}}(\mathbf{y}|H_k)$, and apparently $\sum_{k=1}^{C}\xi_k^N=1$. Because the instants in observation sequence are I.I.D, the whole vector can be segmented into S pieces of successive sub-sequence with corresponding length being N_s, and $N=\sum_{s=1}^{S}N_s$. Furthermore, combine (9) and (12), the ξ_k^N can be written as

$$\xi_k^N=\frac{\prod_{s=1}^{S}\Gamma(\mathbf{y}^{N_s}|H_k)}{\sum_{k'=1}^{C}\prod_{s=1}^{S}\Gamma(\mathbf{y}^{N_s}|H_{k'})}$$
$$=\frac{\prod_{s=1}^{S}\xi_k^{N_s}}{\sum_{k'=1}^{C}\prod_{s=1}^{S}\xi_{k'}^{N_s}}$$
$$=\gamma_k\left(\boldsymbol{\xi}^{N_1},\ldots,\boldsymbol{\xi}^{N_S}\right), \tag{16}$$

where $\boldsymbol{\xi}^{N_s}$ denotes the normalized likelihood vector of segment s. It is meaningful when N is large, and we divide it into several segments and make block calculation. More importantly, (16) indicates that the segment is equivalent to the original classification task, and the design of the block calculation can be flexible and implementable.

III. CNN-Based Classifier

We propose a novel end-to-end DNN enabled AMC [32]–[34], which close approximates the ML-AMC and significantly improves the inference speed.

A. Reasons for Using CNN in AMC

The channel environment is assumed to be time invariant and frequency non-selective, then the elements in observation sequence $\mathbf{y}$ are I.I.D. Some characteristics of $\mathbf{y}$ must be noticed:

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

1) The statistical characteristics of segment $\mathbf{y}^{N_s}$ in arbitrary time interval are the same, i.e.,

$$\mathbb{E}\left\{\mathrm{L}\left(\mathbf{y}_i^{N_s}|H_k,\mathbf{u}_k\right)\right\} = \mathbb{E}\left\{\mathrm{L}\left(\mathbf{y}_j^{N_s}|H_k,\mathbf{u}_k\right)\right\},$$
$$0 \le i,j < N - N_s,$$

where $\mathbf{y}_i^{N_s} = [y_i,\ldots,y_{i+N_s-1}]^T$

2) The signal sequence $\mathbf{y}^N$ can be divided into S segments of successive sub-sequence, and the likelihood function can be expressed as $\mathrm{L}(\mathbf{y}^N|H_k) = \sum_{s=1}^{N_s}\mathrm{L}(\mathbf{y}^{N_s}|H_k)$.
3) Arbitrary exchanges elements in $\mathbf{y}$ for arbitrary times, and the likelihood function of obtained $\mathbf{y}'$ is $\mathrm{L}(\mathbf{y}'|H_k) = \mathrm{L}(\mathbf{y}|H_k)$.

The property 1 indicates that the signal is identically distributed in the time domain. The impulse responses of signal segments at different instants with a convolution kernel are the same, so that the features can be extracted or detected at any time slots. The property 2 proves that the likelihood function is compositional and can be approximated by a stacked cascade representation model. Therefore, a CNN with local receptive fields is more applicable than a full-connected NN. Furthermore, given a large N, the over-fitting problem during training becomes more severe for full-connected models. The property 3 indicates that the signal is sampled as a time series, but it is not time-related, so the recurrent neural network (RNN) is not recommended for this issue.

B. Network Architecture

The illustrative structure of our designed CNN-AMC is shown in Fig. 1, and the input consists of two parts: the preprocessed observation sequence $\mathbf{y}$ with dimension $N = 1000$ and the estimated symbol SNR, which is the same as ML-AMC. However, $\mathbf{y}$ is a raw signal vector which needs to be convoluted to generate higher-level information, while SNR is just a scalar which cannot be convoluted. Different kinds of signals cannot be simply concatenated as the input for the CNN-AMC. Instead, a mixed NN with multiple inputs is employed.

To transform complex vector $\mathbf{y}$ into real data, the real & imaginary parts are separated and then converted into shape $2 \times N$. For example, for the first convolution layer $l = 1$, the input vectors can be denoted by $\mathbf{x}^{l-1} = \mathbf{y} \in \mathbb{R}^{N_{l-1}\times V_{l-1}}$, where the vector number and dimension are $N_{l-1} = 2$ and $V_{l-1} = 1000$, respectively. The sizes of output vectors are listed on the left of corresponding layer boxes in Fig. 1. For a one-dimensional convolution layer l with N_{l-1} input and N_l output vectors, there are $N_l \times N_{l-1}$ convolution kernels (also called as weight matrices) and N_l biases, and the kernel size is denoted by k_l. Giving the output vectors of previous layer $\mathbf{x}_i^{l-1}, i \in \mathcal{N}_{l-1} = \{1,\ldots,N_{l-1}\}$, and the i-th output vector of current layer is represented as

$$\mathbf{x}_i^l = f\left(\sum_{j=1}^{N_{l-1}} \mathbf{x}_j^{l-1} * \mathbf{k}_{i,j}^l + b_i^l\right), \tag{17}$$

where the rectified linear unit (ReLU) [35] is adopted as a fast non-linear activation function: $f(x) = \max(0,x)$, and $*$ is the

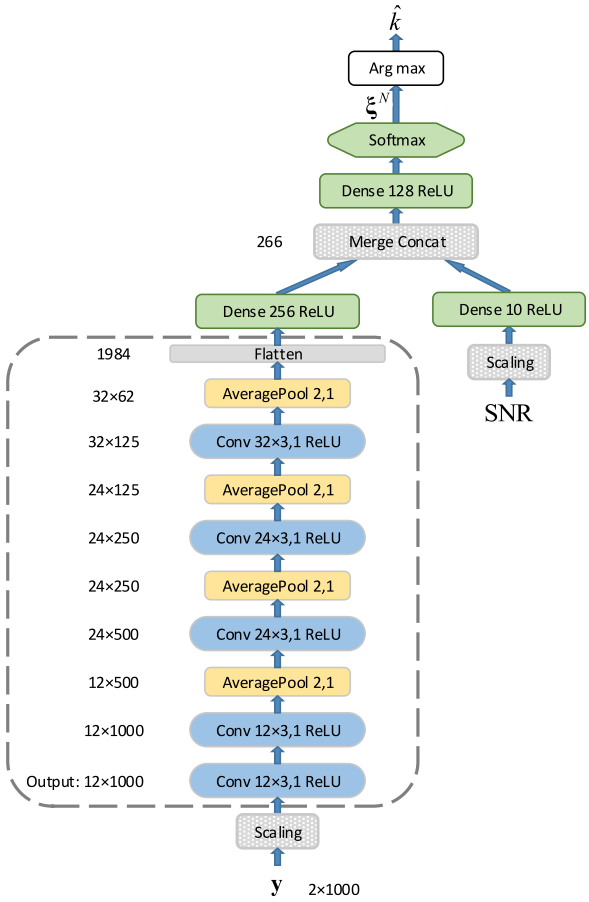

Fig. 1. Illustration of a CNN-based modulation classifier with both vectors and a scalar as input. The text 'Conv 12×3, 1 ReLU' represents that in this convolution layer, the kernel number, kernel size and stride are 12, 3 and 1, respectively. Similarly, the text 'AveragePool 2, 1 represents that in this average pooling layer, window size and stride are 2 and 1, respectively.

convolution operation. There are $N_l \times N_{l-1} \times k_l + N_l$ trainable parameters in convolution layer l. A pooling layer is also called as sub-sampling layer, and it produces down-sampled versions of input vectors. Formally we have the i-th output vector as

$$\mathbf{x}_i^{l+1} = \mathrm{down}\left(\mathbf{x}_i^l\right), \tag{18}$$

where $\mathrm{down}(\cdot)$ represents a sub-sampling function, and the Average pooling [36] is used to reduce the dimensionality by computing the average value of a windowed input vector. After stacked convolution layers and pooling layers, the output vectors are squeezed into a vector by a flatten layer. Then a dense layer is used to encode this vector into a 256-dimensional vector which contains extracted higher-level information. For a dense layer l with input vector $\mathbf{x}^{l-1}$, the output vector can be described as

$$\mathbf{x}^l = f\left(\mathbf{W}^l\mathbf{x}^{l-1} + \mathbf{b}^l\right), \tag{19}$$

where $\mathbf{W}^l \in \mathbb{R}^{N_l\times N_{l-1}}$ and $\mathbf{b}^l \in \mathbb{R}^{N_l}$ are weight matrix and bias, respectively. This is a full-connected layer, and the number of trainable parameters is $N_l \times N_{l-1} + N_l$. Then this vector is concatenated with a vectorized representation of scalar SNR. After another dense layer, then the normalized likelihood vector namely $\hat{\boldsymbol{\xi}}^N$ is produced by a output layer with activation

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

function being softmax. The output $\hat{\boldsymbol{\xi}}^N = [\hat{\xi}_1^N, \ldots, \hat{\xi}_C^N]^T$, and the element $\hat{\xi}_i^N$ is defined as:

$$\hat{\xi}_i^N = \frac{e^{x_i}}{\sum_{j=1}^{C} e^{x_j}}, \tag{20}$$

where x_j is j-th pre-activation output. Finally, the representation of the CNN-AMC can be described as follows

$$\hat{\boldsymbol{\xi}}^N = F(\mathbf{y}, \mathrm{SNR}; \theta), \tag{21}$$

where θ is the network parameter, and $F(\cdot)$ is the overall function of CNN-based classifier. In this deep model, the feature exaction is automatically realized by the CNN part, and the higher-layers mainly deal with the mapping relationship between the input and output data. Generally, the CNN-AMC works as a powerful nonlinear approximator $F(\cdot)$ to the target function (9). The complete training/testing and simulation source codes can be downloaded from https://github.com/mengxiaomao/CNN_AMC.

C. Two-Step Training

In a general training procedure, the CNN-AMC is trained epoch by epoch, and in an epoch, all the training samples are used once to train the CNN-AMC. Within an epoch, the whole training set is shuffled and split into batches with size being N_b, and the model is trained batch by batch. In our paper, the categorical cross-entropy error is adopted as the loss function, and for a training batch the loss function is represented as

$$L(\theta) = -\frac{1}{N_\mathrm{b}} \sum_{i=1}^{N_\mathrm{b}} \boldsymbol{\xi}_i^N \log \hat{\boldsymbol{\xi}}_i^N + \left(1 - \boldsymbol{\xi}_i^N\right) \log\left(1 - \hat{\boldsymbol{\xi}}_i^N\right), \tag{22}$$

where $\boldsymbol{\xi}_i^N$ denotes the target output. This loss function is minimized by the stochastic gradient descent (SGD) algorithm, and the model parameter is updated iteratively as

$$\theta_{t+1} = \theta_t - \eta \nabla L(\theta_t), \tag{23}$$

where η denotes the learning rate, and operation ∇ is the gradient operator. As a variant of SGD algorithm, the adaptive moment estimation (Adam) [37] algorithm is adopted with default settings in our paper. The Adam optimizer dynamically adapts η to accelerate the convergence speed by using adaptive estimates of lower-order momentums, and it is proved to be efficient especially for deep models. The parameter set θ and validation loss at each training epoch are recorded, and the best CNN-AMC is defined as the one with minimal validation loss. Different lengths of received observation sequence under varying SNRs can be generated in the simulation, and the ML-AMC can also produce large amounts of output data.

However, in our experiments, the direct training of the CNN-based classifier is challenging, due to the complicated model and complex target function. With massive amounts of training data and sufficient iteration times, we find that the loss function $L(\theta)$ cannot drop, indicating that this model has stuck in the high-dimensional parameter optimization from the beginning.

Although the final task is difficult, it can be much easier for the model to learn a more straightforward function. There is a simple but useful trick to train the CNN-based classifier by using auxiliary data. As shown in Fig. 2, the whole training process is divided into two steps.

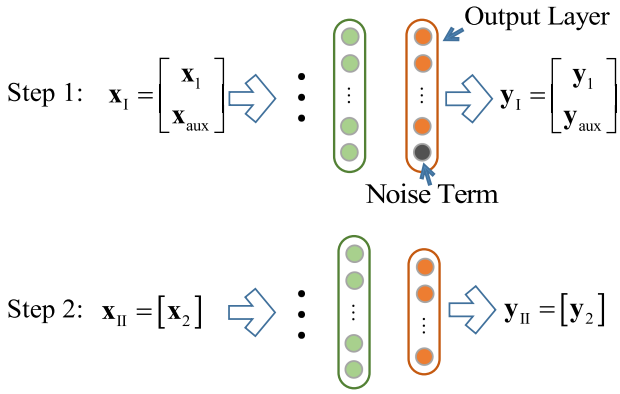

Fig. 2. Illustration of the two-step training of CNN-AMC. In step 1 all the primary output elements are represented with orange circles and the additional noise term is depicted with gray circle. The output element of Gaussian noise is removed in step 2. The input and output in step 1 are denoted by $\mathbf{x}_\mathrm{I}$ and $\mathbf{y}_\mathrm{I}$, and similarly they are marked by $\mathbf{x}_\mathrm{II}$ and $\mathbf{y}_\mathrm{II}$ for step 2.

TABLE I
SIMULATION PARAMETERS OF TWO-STEP TRAINING

Training Step	Step 1	Step 2
Epoch Number	20	240
SNR Range/dB	$[0, 10]$	$[-6, 12]$
Amount of Training Data	79,200	228,060
Amount of Validation Data	8,800	25,340
Using Output of ML Classifier	No	Yes

1) *Pre-training:* This step is designed to help the CNN-AMC converge at the beginning of training, so the scale of the training data is small as shown in Table I. Another output term is added with a gray circle in this step, and its corresponding category is Gaussian white noise with zero means, not other modulation schemes. The amount of auxiliary training samples of noise is the same as any other modulation scheme, and its I/O data are denoted by $\mathbf{x}_\mathrm{aux}$ and $\mathbf{y}_\mathrm{aux}$. The output vectors are set to be binary. In the simulation tests, the loss function $L(\theta)$ decreases, the variance of gradients gets smaller, and the CNN-AMC goes to convergence after some training epochs. The parameter θ with lowest validation loss is stored for the model initialization in the next training.
2) *Fine-tuning:* Firstly, the CNN-AMC loads the stored parameter θ in step 1. The auxiliary output term is redundant in the final classification, so the top layer is replaced with the primary output layer and then randomly initialized. In this step, only the samples of modulation schemes are generated, and the ML-AMC produces the output of training data in the form of normalized likelihood vector $\boldsymbol{\xi}^N$, which is different from the first step. The elaborately designed training data helps the CNN-AMC close approximate the ML-AMC, and simulation result shows that the P_c performance with data generated by ML-AMC has better approximation than the one with common binary labels.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

TABLE II
SUMMARIZATION OF TWO-STEP TRAINING

Step 1:
1) Generate training data, and samples of the additional Gaussian noise is included.
2) Initialize the CNN-AMC parameters randomly.
3) Train CNN-AMC, store the model parameters with the minimal validation loss.
Step 2:
1) Produce training data, and the target output is achieved from ML-AMC.
2) Replace the top layer of CNN-AMC and randomly initialize it; Load the other model parameters from storage.
3) Train CNN-AMC, store the model parameters with the minimal validation loss.

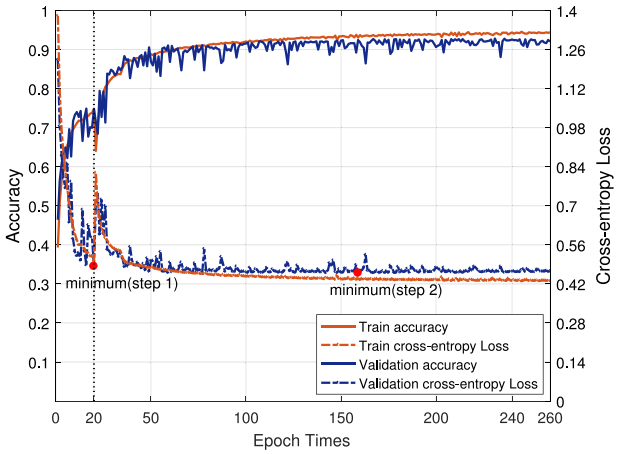

Fig. 3. Training/validation curves of accuracy and loss. The x-axis denotes the training epoch. The validation and training curves are plotted in dark blue and orange lines, and the accuracy and loss curves are depicted with solid and dotted lines.

Similarly, the CNN-AMC with lowest validation loss is stored, reloaded and tested in the following simulations, and the detailed settings are listed in Table I.

The summarization of the two-step training method is given in Table II. In Fig. 3, the curves of training/validation loss/accuracy in a complete two-step training process are plotted. Generally, the loss curves go down, and the accuracy curves keep rising in both two steps. In step 2, the training loss drops sharply at the beginning and then descends steadily. The curve of validation loss has the similar trend as the training loss one. While as the epochs increasing it goes stable with small jitters, and it is hard to be less than 0.47, which means the training is in the saturation region. The corresponding θ with the minimal validation loss is stored, and the minimum value in step 2 is labeled with the red dot.

The likelihood function of the auxiliary Gaussian noise in the complex plane is quasi-convex, which is a relatively more straightforward function to approximate. From (7) we know that the conditional PDF of one observation instant is a weighted sum of Gaussian distribution with different means and same variance. Therefore, the approximation to more complex PDFs of other modulation schemes becomes much more comfortable for a CNN-AMC which has learned the function of Gaussian noise. After learning the approximation of a symbol, then the objective function becomes more accessible according to (9). In the first step, the problem of initial convergence is solved. By pre-training, the CNN-AMC is further trained to approximate the optimal ML-AMC as close as possible in the second step.

D. Transfer Learning

The fore-mentioned two-step training deals with the problem: Given a CNN-AMC with random parameter initialization, and how to train it to classify the modulation scheme with given training and validation data. In fact, one learned CNN-AMC can be applied to one specific task such as the coherent environment with several candidate modulation modes. Once the scenario or the modulation set is changed, the CNN-AMC needs to be retrained with new training data.

In fact, it is not necessary to restart the new training process from the very beginning with the aid of transfer learning [38]. As the paper states before, the CNN part of the CNN-AMC generates higher-level information from the preprocessed raw observation sequence. The feature extraction can be employed universally in varying tasks, and the training can be much easier with some already learned prior knowledge or experiences instead of purely blank. Here, we take two cases as examples:

1) *Diverse modulation schemes:* In distinct situations, some modulation modes are added or removed, depending on the requirement of the classification task. Except for the new training data, the output layer of the CNN-AMC needs to be changed to the new corresponding category. Therefore, all the CNN-AMC parameters except for the ones in the top layer can be loaded as the initialization instead of randomly setting.
2) *Different environments:* During the preprocessing period, the carrier phase compensation is included in the coherent case but not incoherent scenario. The CNN-AMC can be trained with new incoherent training data using the proposed two-step method. Moreover, the CNN-based classifier learned in the coherent case can be utilized as the initial model for training with these data and all the model parameters are loaded from the trained one in coherent case. We give an example using transfer learning, and its counterpart is the one using two-step training. As shown in Fig. 4, descent speed of loss curve is faster than that using two-step training. Besides, the pre-training step can be left out, which further reduce the time cost of the training period.

Although transfer learning helps to improve the training efficiency, it is found by two-step learning.

E. Unit Classifier

To achieve high classification accuracy with signals at low SNR, the sequence space N can be up to several or even tens of thousands. More training samples are needed to overcome overfitting as N increases, and thus it makes the training much more difficult. The problem is that with limited memory space and computation resources, the classification and its corresponding

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

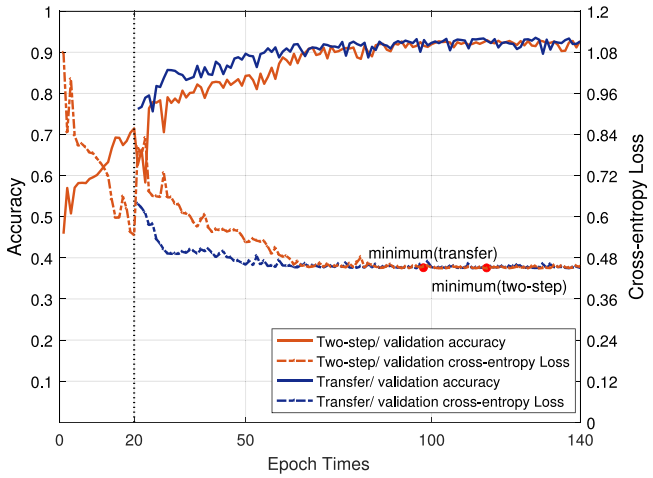

Fig. 4. Loss and accuracy curves of validation data versus epoch times. The one using transfer learning takes less time to reach the saturation region than the two-step training one, and the pre-training step is reduced.

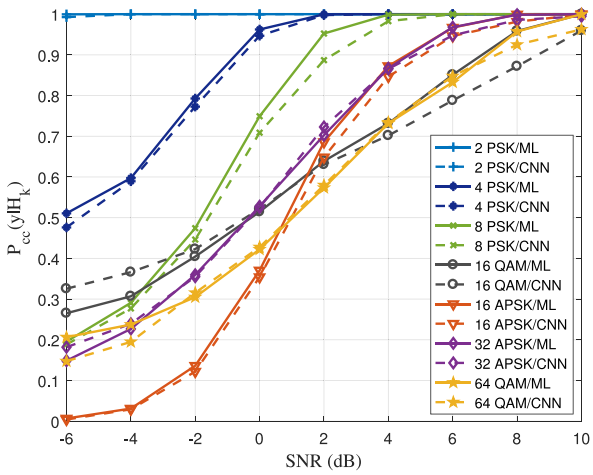

Fig. 5. $P_{\rm cc}\left(\mathbf{y}|H_k\right)$ curves of 7 modulation modes with SNR range being $[-6, 10]$ dB. The ML and CNN-based classifiers are depicted with solid and dotted lines, respectively.

training with a large N can be hard or even impractical. On the other hand, it is also challenging to design a classifier with changeable input space.

According to the previous analysis in Subsection II-C, the input $\mathbf{y}$ of a large N can be divided into segments with unit length. The CNN-based classifier with the input signal $\mathbf{y}$ of unit input dimension is defined as a unit classifier. Theoretically, the segment has the same classification performance as the complete observation one in coherent environment. More importantly, the scales of the unit classifier regarding training, cache, and calculation are adjustable. Hence, the design of the AMC can be flexible to suit the hardware, and it is irrelevant to the sequence space N. The inference can also be made parallelly for all unit classifiers.

IV. Simulation Results

The common sequence space N for AMC varies from hundreds to thousands through our investigation, and considering the hardware condition, the CNN-AMC with observation space $N = 500$ (named CNN-500) and $N = 1000$ (named CNN-1000) are tested in this section. Besides, the ML-AMC and a feature-based AMC using 9 high-order cumulants [39] (named Cu-AMC) are served as the comparisons. The second order, fourth order and sixth order cumulants are adopted as features, and a two-hidden-layer NN is used as a classifier. Couples of modulation schemes are considered. The overall and conditional correct probabilities versus SNR are the two main issues needed to be studied. Besides, many other aspects must be considered such as robustness against estimation error, inference time cost, and system storage requirement.

A. Coherent

The simulations begin with the coherent environments, and seven modulation modes namely 2PSK, 4PSK, 8PSK, 16QAM, 16APSK, 32APSK and 64QAM with equal prior probabilities are considered. The receiver is assumed to have precise knowledge of parameter set $\mathbf{u}_k$ including the constant phase offset θ_c and SNR, but the modulation scheme needs to be inferred. Apart from the power normalization by (6), the observation sequence is compensated with the conjugate phase offset in the preprocessing period

$$\mathbf{y} = \mathbf{y}e^{-j\theta_c}. \tag{24}$$

Combine (7), (8) and (9), the likelihood function under hypothesis H_k is explicitly given by

$$\Gamma(\mathbf{y}|H_k) \propto \frac{1}{M_k}\prod_{n=0}^{N-1}\sum_{i=1}^{M_k}\exp\left(\frac{|x_n^{k,i}|^2 - 2\Re\mathfrak{e}\{y_n^* x_n^{k,i}\}}{N_0}\right). \tag{25}$$

Hence, the normalized likelihood vector $\boldsymbol{\xi}^N$ can be obtained, and tied with observation sequence $\mathbf{y}$, plenty of training and validation data can be generated for the CNN-AMC.

1) Correct Probability: The deep learning enabled AMC is greatly faster than the ML-AMC. While as an approximation to the optimal AMC, it can suffer some performance degradation in terms of $P_{\rm cc}\left(\mathbf{y}|H_k\right)$ and $P_{\rm c}$, which are studied in this simulation.

Firstly, the conditional correct probabilities $P_{\rm cc}\left(\mathbf{y}|H_k\right)$ curves of 7 modulations are illustrated in Fig. 5 ($N = 1000$). The ML-AMC are presented with solid lines, and the CNN-1000 ones are plotted with dashed lines. Clearly 2PSK is the most easily recognized modulation in our set, and its conditional correct probability is 100% even at the region of low SNR. The 4PSK and 8PSK are also found to be more easily identified compared to the other modulation schemes, and their $P_{\rm cc}\left(\mathbf{y}|H_k\right)$ is up to 1.00 when SNR $= 2$ dB and SNR $= 4$ dB, respectively. In high SNR field (SNR $= 10$ dB), all the modulation schemes can be recognized almost without any error for ML-AMC.

In general, there is a little $P_{\rm cc}\left(\mathbf{y}|H_k\right)$ performance loss when comparing the CNN-1000 to the ML-AMC. At some SNR regions, for some modulations the $P_{\rm cc}\left(\mathbf{y}|H_k\right)$ are higher than the ML-AMC when using CNN-1000. For example, in the case where SNR ranging from -6 to -2 dB and the modulation scheme being 16 QAM, the CNN-1000 seems better than ML-

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

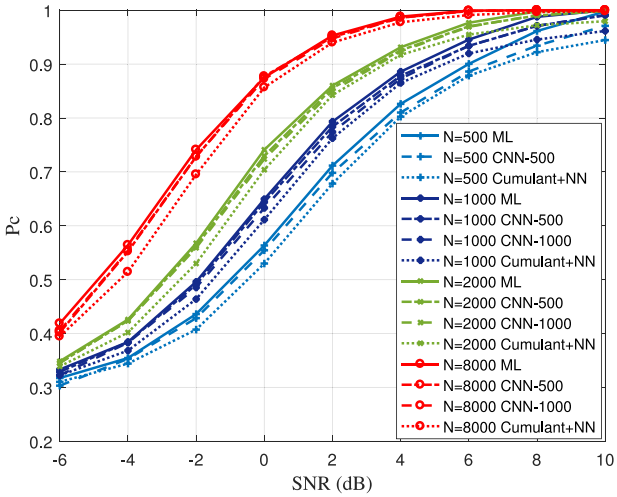

Fig. 6. P_c curves of different sequence space N versus SNR. The ML-AMC, CNN-500 and CNN-1000 are depicted with solid, dotted and dashed lines, respectively.

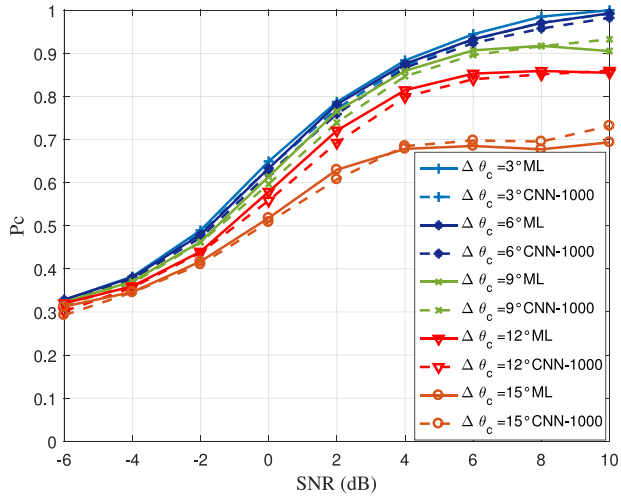

Fig. 7. P_c curves of different estimation error on phase offset $|\Delta\theta_c|$ in set $\{3^\circ, 6^\circ, 9^\circ, 12^\circ, 15^\circ\}$. The ML and the CNN-1000 classifiers are depicted with solid and dashed lines, respectively.

AMC. This is because the corresponding conditional probability is overestimated, at the cost of a lower correct classification probability of some other modulation scheme such as 64QAM. The P_c of CNN-1000 can only infinitely approximate the ML-AMC, and it cannot exceed the optimal one.

The curves of the overall correct probabilities for ML-AMC, CNN-500, CNN-1000 and Cu-AMC are presented in Fig. 6. There is about 1% to 2% P_c performance degradation between the CNN-AMC and the ML-AMC when $N = 500$ and $N = 1000$. Moreover, the CNN-500 and CNN-1000 are used as unit classifier proposed in III-E when sequence space N is integer multiples of the unit input length, and the approximation is almost very near the curves of ML-AMC when $N = 2000$ and $N = 8000$. The simulation results show that under the precise knowledge of the parameter set, the CNN-based classifiers can closely approximate the ML-AMC. On the other hand, its calculating speed is about 150× to 170× faster than ML enabled approach with given 7 modulation schemes in coherent scenario. Besides, the CNN-based classifiers outperform the Cu-AMC, and the feature-based method works worse than CNN-AMCs when $P_c > 90\%$. More importantly, the result indicates that the deep learning enabled approach has better performance than the manual feature extraction-based method in this case.

2) Estimation Error: The previous simulations are given under the assumption of perfect knowledge of two parameters: fixed phase offset and SNR. However, the estimation error is usually inevitable. In this part, simulations are conducted with varying degrees of estimation error, and the variable-controlling approach is adopted.

In the first simulation the deviation in carrier phase is investigated. Because of the priori knowledge that common constellation diagrams are symmetric, the influence of positive and negative deviation on carrier phase is equivalent. We use $|\Delta\theta_c|$ denote phase error and the $|\Delta\theta_c| \in \{3^\circ, 6^\circ, 9^\circ, 12^\circ, 15^\circ\}$.

The result is shown in Fig. 7. Generally speaking, the P_c decreases as the $|\Delta\theta_c|$ rises. When $|\Delta\theta_c| < 6^\circ$ the classification performance degradation is very small. The recognition has

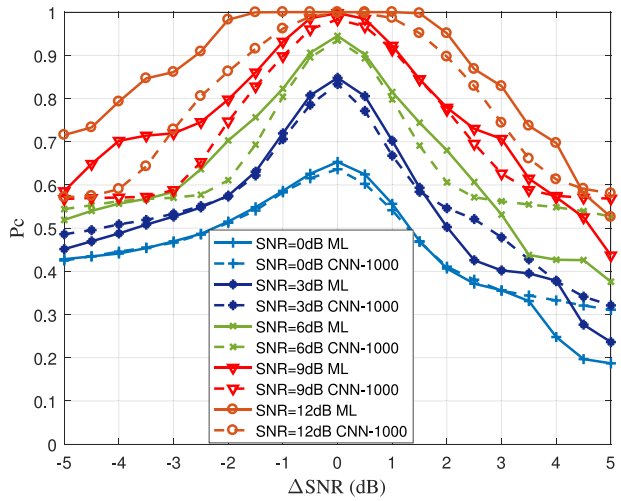

Fig. 8. P_c curves of different SNR in set $\{0, 3, 6, 9, 12\}$ with estimation error ΔSNR range being $[-5, 5]$ dB. The ML-AMC and the CNN-1000 are depicted with solid and dashed lines, respectively.

obviously deteriorated when $|\Delta\theta_c| \in [6^\circ, 9^\circ]$, and the ML-AMC gets even worse when SNR increases at the region of $[8, 10]$ dB ($|\Delta\theta_c| = 9^\circ$). The P_c remains about 0.86 when SNR > 6 dB with $|\Delta\theta_c| = 12^\circ$. The P_c cannot even exceed 0.7 when $|\Delta\theta_c| = 15^\circ$. As comparison the CNN-based classifier is about the same as the ML-AMC, and in some cases it can perform better. In the extreme case with $|\Delta\theta_c|$ being up to 15°, the P_c of CNN-based classifier is higher than ML-AMC, which means the NN enabled approach is more robust.

In our second simulation, the influence of SNR estimation error is studied. The mistake can lead to an erroneous assumption of concentrations of data around the constellation points, and the centers of these alphabets are also normalized with a coefficient with deviation.

As shown in Fig. 8, the x-axis is the SNR error ranging from −5 to 5 dB, and it is defined as $\Delta\text{SNR} = \widehat{\text{SNR}} - \text{SNR}$, and the y-axis is the P_c. Curves of real SNR $\in \{0, 3, 6, 9, 12\}$ dB are

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

labeled with different colors and markers. It can be seen that the classifiers are sensitive to the SNR error, even 0.5 dB estimation bias can bring about the inference degradation. Besides, an overestimated SNR is more harmful than an underestimated SNR for the classifier. We also notice that curves of CNN-AMC is smoother than the ML-AMC, and in many cases, the difference in P_c between the ML-AMC and CNN-based classifier can be larger than 10%. At the region of SNR $= 12 \pm 1.5$ dB, the P_c of ML-AMC remains 1.0, which indicates that the performance saturation can be against SNR estimation error to some extent. Meanwhile, the corresponding CNN-based classifier has the worst performance in all 5 comparisons. We suppose that this degradation can be attributed to inadequate training data. In Table I the training data is sampled in the range $[-6, 12]$ dB, which means the CNN-AMC is not trained including this range.

One feasible approach to make the classifier robust to the SNR error is to train the CNN-AMC with adding noise on SNR. This method can make the classifier less sensitive to the deviation on SNR, but at the expense of overall classification performance.

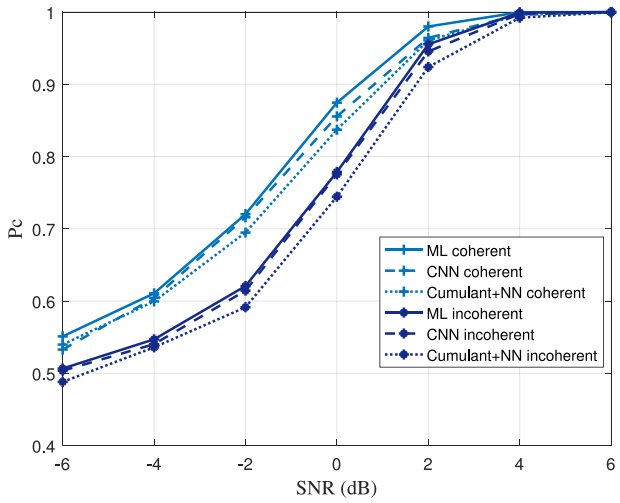

Fig. 9. P_c curves of ML-AMC and CNN-500 versus SNR in range $[-6, 6]$ dB. Classifiers in coherent case are plotted in light blue lines, and depicted with navy blue lines in incoherent scenario.

B. Incoherent

The coherent environment is usually not accessible in non-cooperative case, and the blind phase estimation algorithm is required at the receiver. In the previous simulation classification performance of the CNN-AMC is greatly influenced when phase estimation error $|\Delta\theta_c|$ is larger than 6°, so the final classification accuracy highly depends on the employed estimation algorithms. In this subsection, the θ_c is assumed to be a random variable which is uniformly distributed in range $[0, 2\pi)$. Similarly, the likelihood function can be obtained as follows

$$\begin{aligned}\Gamma(\mathbf{y}|H_k) &\propto \mathbb{E}_{\theta_c}\left\{\frac{1}{M_k}\prod_{n=0}^{N-1}\sum_{i=1}^{M_k}\exp\left(-\frac{|x_n^{k,i}|^2}{N_0}\right)\right.\\ &\quad\left.\times\exp\left(\frac{2\mathfrak{Re}\{y_n^* x_n^{k,i} e^{j\theta_c}\}}{N_0}\right)\right\}\\ &= \frac{1}{M_k}\int_0^{2\pi}\prod_{n=1}^{N}\sum_{i=1}^{M_k}\left[\exp\left(-\frac{|x^{k,i}|^2}{N_0}\right)\right.\\ &\quad\left.\times\exp\left(\frac{2\mathfrak{Re}\{y_n^* x^{k,i} e^{j\theta_c}\}}{N_0}\right)\right]d\theta_c. \qquad (26)\end{aligned}$$

In fact, the first Bessel function of zero order can be utilized to simplify (26), but it is not suggested in numerical calculation due to its complex nature. In replace, the expectation over θ_c can be adequately approximated with the help of composite Simpson's rule [20] by splitting the integration interval $[0, 2\pi)$ into subintervals of same lengths. The range can be shorted to $[0, \pi)$, taking the use of constellation points symmetry in the complex plane. The additional integration over θ_c makes the total amount of calculations is several times of that in coherent environment, and the computation time cost is proportional to the number of subintervals.

To reduce the total amount of calculations, only the 2PSK, 4PSK, 8PSK and 16QAM are included in the modulation set. Besides, only the CNN-500 is retrained with incoherent data of observation space $N = 500$. In the preprocessing period, the phase compensation step in (24) is not needed.

1) Coherent VS Incoherent: Both the overall probabilities of correct classification with ML-AMC, CNN-AMC and Cu-AMC are plotted in Fig. 9. It can be seen that under the incoherent scenario, the approximation of CNN-AMC is very close to the optimal one, and it also outperforms the Cu-AMC. As a comparison, the overall correct probabilities of three classifiers in coherent scenario are illustrated with light blue lines. The P_c performance of CNN-AMC is a little better than that of the Cu-AMC in coherent case, but its advantage is enlarged in the incoherent scenario. To achieve the $P_c = 90\%$, the SNR required for the incoherent classifier is up to 1.6 dB, which is about 1.0 dB more than the coherent counterpart. Generally, the gap between the performance of the coherent and incoherent classifiers is gradually narrowed as the SNR increases. The correct decision probabilities of both classifiers are all at the point of 100% when SNR $= 4$ dB.

The incoherent CNN-500 is also tested with observation space $N \in \{500, 1000, 2000\}$, and the ML-AMC is treated as the optimal contrast in Fig. 10 with dotted lines. The text 'slicing' indicates that the unit classifier is used to calculate the likelihood function, and the ML-AMC using segment is depicted with solid lines. Theoretically, the block calculation of observation signal is not optimal in the incoherent environment. From the illustration it can be seen there is about 0.2 dB and 0.4 dB P_c loss with the segment number N_s being 2 and 4. On the other hand, the approximation of unit CNN-500 is also very near the ones of ML-AMC using segment with different input space. When $N \in \{1000, 2000\}$, the overall correct probabilities of classifiers are up to 100% at the region of SNR $= 2$ dB. Although the unit classifier suffers some performance loss, it is feasible as a sub-optimal approach in incoherent scenario. In addition, new CNN-AMCs with specific input space can be trained as an approximation to the optimal method.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

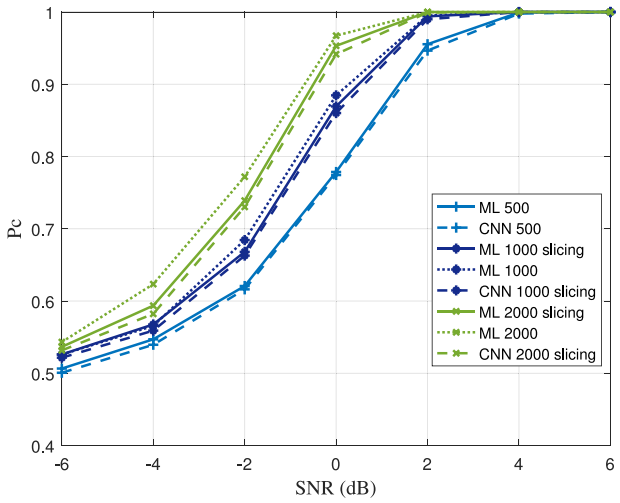

Fig. 10. P_c curves of ML-AMC and CNN-500 versus SNR with varying observation sequence space N. The ML-AMCs are plotted with dotted lines, and illustrated with solid lines when signal segment is used. The CNN-AMCs are depicted with dashed lines, and the unit classifiers are employed.

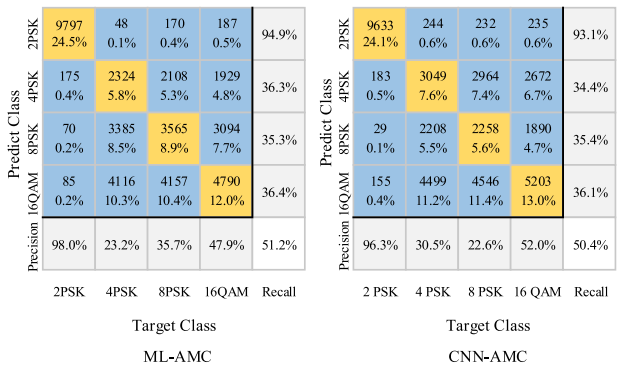

Fig. 11. The visualized confusion matrices of ML-AMC on the left side and CNN-AMC on the right side at the region of SNR $= -6$ dB. Each target class is tested with 10000 samples, and the exact number and its proportion are given in blue and yellow blocks. In the gray column block, the gray row block and the white block, the values of recall, precision and accuracy are given, respectively.

2) Conditional Correct Probability: Through initial analysis on results of four conditional correct probabilities using ML-AMC and CNN-AMC, we find that the approximations of the CNN-based classifiers are quite close to ML-AMC in high SNR region, but they do not match well when SNR $<$ 2dB.

To further study the conditional correct probability performance, the confusion matrices of two AMCs are visualized in Fig. 11 at the region of SNR $= -6$ dB. Each row represents the instances in a predicted class of AMC, and each column denotes the instances in a target modulation class. Comparing the recall and precision of two AMCs, the main difference is on 4PSK and 8PSK. Both the 4PSK and 8PSK are phase shift keying, and the distance between adjacent points is small in the complex plane. Therefore, these two kinds of modulation schemes are easily mistakenly classified for each other, especially in low SNR field. The loss function is set to maximize the overall correct probability, and even though the P_c is close approximated in Fig. 9, the approximation of conditional correct probabilities cannot be guaranteed in this case.

C. Flat Fading Channel

The non-frequency selective slow-fading channel is studied in this part. Therefore, the amplitude a and carrier phase θ_c are assumed be constants over the whole observation period. The PDF of carrier phase is the same as the one in IV-B1. Besides, the PDF of amplitude a is assumed to follow the Rayleigh distribution which is given by

$$p(a) = \frac{2a}{\sigma_a^2} \exp\left(-\frac{a^2}{\sigma_a^2}\right), \qquad a \geq 0 \tag{27}$$

where $\sigma_a^2 = \mathbb{E}[a^2]$. Except for the preprocessed observation sequence $\mathbf{y}$, the knowledge of symbol SNR is still required for the CNN-AMC. The likelihood function is represented as

$$\begin{aligned}\Gamma(\mathbf{y}|H_k, a) &\propto \mathbb{E}_{\theta_c}\left\{\frac{1}{M_k}\prod_{n=1}^{N}\sum_{i=1}^{M_k}\exp\left(-\frac{|ax^{k,i}|^2}{N_0}\right)\right. \\ &\qquad \left.\times \exp\left(\frac{2\mathfrak{Re}\{y_n^* a x^{k,i} e^{j\theta_c}\}}{N_0}\right)\right\} \\ &= \frac{1}{M_k}\int_0^{\pi}\prod_{n=1}^{N}\sum_{i=1}^{M_k}\left[\exp\left(-\frac{|ax^{k,i}|^2}{N_0}\right)\right. \\ &\qquad \left.\times \exp\left(\frac{2\mathfrak{Re}\{y_n^* a x^{k,i} e^{j\theta_c}\}}{N_0}\right)\right] d\theta_c.\end{aligned} \tag{28}$$

The single input and multiple output (SIMO) case is considered in the simulation, where there is one transmitter and one receiver with N_a antennas. The channels are assumed to be independent to each other. The likelihood function with multiple observation is given as

$$\Gamma\left(\{\mathbf{y}_1, \ldots, \mathbf{y}_{N_a}\}|H_k, \{a_1, \ldots, a_{N_a}\}\right) = \prod_{u=1}^{N_a}\Gamma(\mathbf{y}_u|H_k, a_u), \tag{29}$$

which ignores the fact that the transmitted symbol sequence is the same to all antennas. Formally, the process of multiple observation of one source is similar to the signal segment in (16) for CNN-AMC.

The receiver using $1, 2, 4$ antennas is investigated in our simulation, and thus the overall classification performance is tested versus average symbol SNR, which is denoted by $\overline{\text{SNR}}$. As shown in Fig. 12, the curve of CNN-500 is illustrated and the ML-AMC is treated as its counterpart. It can be seen that in flat Raleigh fading channel, the receiver with 1 antenna reaches the overall correct probability $P_c = 90\%$ when $\overline{\text{SNR}} = 4$ dB. In contrast, the $\overline{\text{SNR}}$ required for the receiver with 2 and 4 antennas are about -1 dB and -3 dB, respectively. When $\overline{\text{SNR}} = 2$ dB, there is almost no classification error in the case with 4 antennas. Combining with the performance in Fig. 6 and Fig. 9, the performance of CNN-AMC can generally achieve

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

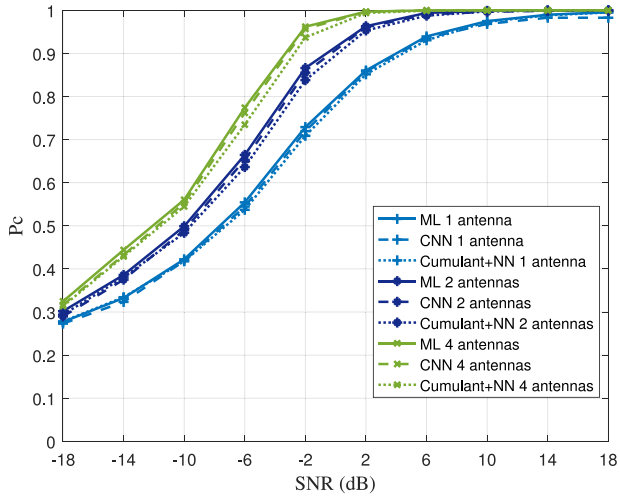

Fig. 12. Plots of P_c versus $\overline{\text{SNR}}$ in range being $[-18, 18]$ dB. The observation space $N = 500$ for each antenna, and the curves of $1, 2, 4$ antennas are illustrated with light blue, navy blue and green lines, respectively.

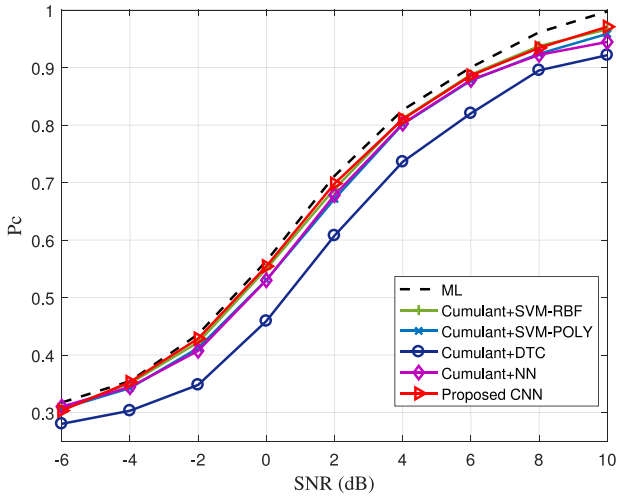

Fig. 13. Baseline test for various AMCs in the coherent environment ($N = 500$). Plots of P_c versus SNR.

a little closer approximation to the optimal one, in comparison with Cu-AMC.

D. Feature-Based Methods

Except for the aforementioned NN-based method, some other classifiers are considered in this subsection, including decision tree classifier (DTC), SVM with polynomial and radial basis function (RBF) kernels, i.e., SVM-POLY and SVM-RBF, respectively. The dataset ($N = 500$) in IV-A is served as the baseline test in the coherent environment. As shown in Fig. 13, the overall P_c of CNN-AMC and SVM-RBF is very similar, and the NN-based AMC suffers a little P_c performance loss compared to these two. the DTC classifier has the lowest classification correct rate.

Then these AMCs are tested with the incoherent baseline dataset ($N = 500$) in IV-B. Combining with Fig. 13, the correct

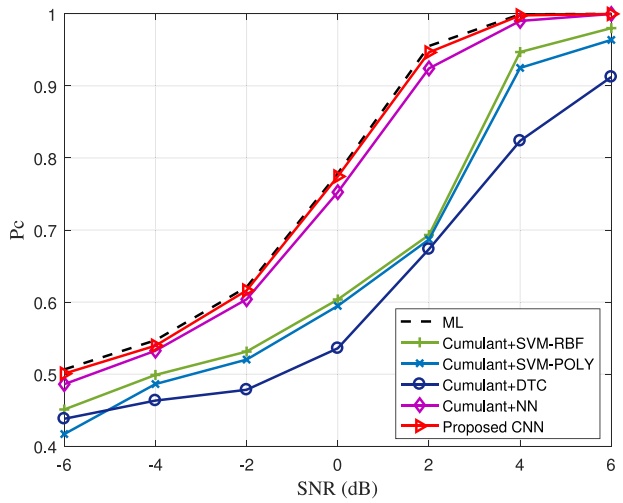

Fig. 14. Baseline test for various AMCs in the incoherent environment ($N = 500$). Plots of P_c versus SNR.

TABLE III
AVERAGE INFERENCE TIME COST OF CLASSIFIERS IN COHERENT ENVIRONMENT

N	500	1000	2000	8000
ML-AMC/ms	10.8	21.3	42.6	164.5
CNN-500/ms	0.0727	0.146	0.288	1.15
CNN-1000/ms	-	0.127	0.246	1.02

classification performance of DTC classifier is still the worst, and the SVM-based methods are not stable in Fig. 14. Among the feature-based methods, only the NN-based AMC can achieve the closest P_c performance to the CNN-AMC in both coherent and incoherent environments, and that is why it is served as the Cu-AMC in previous simulations. Besides, the P_c performance of CNN-AMC is a little better than that of NN-based AMC.

E. Complexity & Implement

In this subsection, we give a discussion on the complexity analysis and implement of ML-AMC and CNN-AMC. The ML-AMC requires no memory, and the scales of the proposed CNN-500 and CNN-1000 are small, and they occupy only 3.4 MB and 6.3 MB capacity with $N = 500$ and $N = 1000$, respectively. The simulation platform is presented as: cpu intel i7-6700k and gpu nvidia GTX-960. The ML-AMC is implemented with Matlab and mainly the cpu is utilized. While the CNN-AMC runs on the Python platform with the gpu.

1) Coherent: Table III lists the time cost per inference with different input space of three classifiers in IV-A. The unit time cost is defined as time cost per inference divided by N, and they are 20.8 μs, 0.144 μs and 0.127 μs for ML-AMC, CNN-500 and CNN-1000, respectively. The calculating speed is accelerated by more than two orders of magnitude with NN-based approach in this specific case. More concretely, the CNN-500 and CNN-1000 are about 150 and 170 times faster than ML-AMC, respectively.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
AVERAGE TIME COST OF CLASSIFICATION OF FOUR MODULATION SCHEMES

Coherent	N	500	1000	2000
	ML-AMC/ms	3.08	6.07	11.8
	CNN-500/ms	0.0775	0.155	0.309
Incoherent	N	500	1000	2000
	ML-AMC/ms	138	271	528
	CNN-500/ms	0.0775	0.155	0.309

TABLE V
COMPLEXITY OF ML-AMC AND CNN-AMC UNDER THREE SCENARIOS

	ML-AMC	CNN-AMC
Coherent	$k_{\text{ml}}O(N\sum_{k=1}^{C}M_k)$	$k_{\text{cnn}}O(N)$
Incoherent	$k_{\text{ml}}O(NN_{\theta_c}\sum_{k=1}^{C}M_k)$	$k_{\text{cnn}}O(N)$
Flat fading channel	$k_{\text{ml}}O(NN_{\theta_c}N_a\sum_{k=1}^{C}M_k)$	$k_{\text{cnn}}O(NN_\alpha)$

2) Incoherent: In incoherent cases, the additional integration on carrier phase is required for ML-AMC. Theoretically, the unit time cost is about N_{θ_c} times of that in coherent scenario, where N_{θ_c} is the number of subintervals. After simulation tests in IV-B1, the unit time costs listed in Table IV are 6.01 μs and 268 μs in coherent and incoherent scenarios, respectively. The latter one is about 44.7 times of the coherent one, which is approximately equal to the adopted variable value $N_{\theta_c} = 45$.

On the other hand, the unit time cost of CNN enabled approach is only related to the architecture of the model. Except for the top layer, the structure of the CNN-500 for seven and four modulation schemes are the same, so the average time cost of CNN-500 in Table IV is very similar to the corresponding ones in Table III. Besides, the average time cost of CNN-based approach is irrelevant to the communication environment, and its unit time cost is 0.155 μs. Therefore, in this simulation case the CNN-500 is about 40 and 1700 times faster than ML-AMC in coherent and incoherent environment, respectively.

Both the amount of calculations grows linearly as observation space N increases for ML-AMC and CNN-AMC. The computational complexity is mainly determined by network architecture for our proposed CNN-AMC, and the modulation schemes and channel environments has negligible influence on the total computation. In contrast, the ML-AMC is greatly affected by these issues. The computational complexities of two AMCs in our three simulated environments are listed in Table V. The notations k_{ml} and k_{cnn} are the coefficients of corresponding AMCs. According to Table V, intuitively we can see that the computational complexity of CNN-AMC is less than ML-AMC. In fact, the coefficient of CNN-AMC k_{cnn} is much larger than that of ML-AMC, whose total amount of calculations can be smaller in the coherent scenario. However, the time cost per inference of CNN-AMC can be sharply reduced in implement, attributing to the parallel processing. The output of different kernels or neurons of a layer can be achieved parallelly, and also the classifications of different received sequences can be processed simultaneously. On the other hand, the calculation complexity of ML-AMC can be easily affected by the channel environments and also the modulation set. In contrast, the time cost per inference of proposed CNN-AMC is unchanged, working as a deep learning model with powerful and redundant approximation ability. This indicates that the CNN-AMC can outperform the ML-AMC especially in more complex environments.

V. CONCLUSION

Automatic modulation classification enabled by deep learning approach has been proposed in this paper. As a parallel computing model, a deep learning based classifier named CNN-AMC is proposed as an approximation to the ML-AMC. The CNN-AMC automatically extracts features from the long symbol-rate sequence, and the SNR is also required as input. The corresponding two-step training process and transfer learning are introduced to solve the challenging training task and improve retraining efficiency, respectively. Besides, a unit classifier is also proposed to suit varying observation sequence dimensions flexibly. The simulation results show that the CNN-based classifier can outperform the feature-based methods, and obtain the closest approximation to the optimal ML-AMC. Besides, CNN-AMCs have the certain robustness to estimation error on carrier phase offset and SNR. By taking advantage of parallel computation, the deep learning-based approach is about 40 to 1700 times faster than the ML-AMC regarding average classification speed, depending on the given scenarios. In our future work, we will focus on the robustness of CNN-based classifiers to varying deviations, and applications in complicated non-cooperative scenarios will also be investigated.

REFERENCES

[1] E. E. Azzouz and A. K. Nandi, "Automatic modulation recognition of communication signals," *IEEE Trans. Commun.*, vol. 334, no. 4, pp. 431–436, Apr. 1998.
[2] O. A. Dobre, A. Abdi, Y. Bar-Ness, and W. Su, "Survey of automatic modulation classification techniques: Classical approaches and new trends," *IET Commun.*, vol. 1, no. 2, pp. 137–156, Apr. 2007.
[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer-Verlag, 2006.
[4] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Commun. Surveys Tut.*, vol. 15, no. 3, pp. 1136–1159, Jul.–Sep. 2013.
[5] A. Hazza, M. Shoaib, S. A. Alshebeili, and A. Fahad, "An overview of feature-based methods for digital modulation classification," in *Proc. Int. Conf. Commun., Signal Process., Appl.*, 2013, pp. 1–6.
[6] S. U. Pawar and J. F. Doherty, "Modulation recognition in continuous phase modulation using approximate entropy," *IEEE Trans. Inf. Forensics Secur.*, vol. 6, no. 3, pp. 843–852, Sep. 2011.
[7] Y. C. Deng and Z. G. Wang, "Modulation recognition of MAPSK signals using template matching," *Electron. Lett.*, vol. 50, no. 25, pp. 1986–1988, 2014.
[8] L. Xie and Q. Wan, "Cyclic feature based modulation recognition using compressive sensing," *IEEE Wireless Commun. Lett.*, vol. 6, no. 3, pp. 402–405, Jun. 2017.
[9] A. Swami and B. M. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Trans. Commun.*, vol. 48, no. 3, pp. 416–429, Mar. 2000.
[10] H. C. Wu, M. Saquib, and Z. Yun, "Novel automatic modulation classification using cumulant features for communications via multipath channels," *IEEE Trans. Wireless Commun.*, vol. 7, no. 8, pp. 3098–3105, Aug. 2008.
[11] S. Aaron, E. Michael, and D. Joseph, "Modulation classification of satellite communication signals using cumulants and neural networks," in *Proc. Cogn. Commun. Aerosp. Appl. Workshop*, 2017, pp. 1–8.
[12] C. S. Park, J. H. Choi, S. P. Nah, W. Jang, and D. Y. Kim, "Automatic modulation recognition of digital signals using wavelet features and SVM," in *Proc. Int. Conf. Adv. Commun. Technol.*, 2008, pp. 387–390.
[13] D. Boutte and B. Santhanam, "A hybrid ICA-SVM approach to continuous phase modulation recognition," *IEEE Signal Process. Lett.*, vol. 16, no. 5, pp. 402–405, May 2009.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.

[14] Z. Zhang, Y. Li, X. Zhu, and Y. Lin, "A method for modulation recognition based on entropy features and random forest," in *Proc. IEEE Int. Conf. Softw. Quality, Rel. Secur. Companion*, 2017, pp. 243–246.

[15] M. L. D. Wong and A. K. Nandi, "Automatic digital modulation recognition using artificial neural network and genetic algorithm," *Signal Process.*, vol. 84, no. 2, pp. 351–365, 2004.

[16] A. K. Nandi and E. E. Azzouz, "Modulation recognition using artificial neural networks," *Signal Process.*, vol. 56, no. 2, pp. 165–175, 1996.

[17] D. Wang *et al.*, "Combatting nonlinear phase noise in coherent optical systems with an optimized decision processor based on machine learning," *Opt. Commun.*, vol. 369, pp. 199–208, 2016.

[18] X. Zhang, J. Chen, and Z. Sun, "Modulation recognition of communication signals based on SCHKS-SSVM," *J. Syst. Eng. Electron.*, vol. 28, no. 4, pp. 627–633, 2017.

[19] J. L. Xu, "Likelihood-ratio approaches to automatic modulation classification," *IEEE Trans. Syst. Man Cybern. C*, vol. 41, no. 4, pp. 455–469, Jul. 2011.

[20] J. L. Xu, W. Su, and M. C. Zhou, "Software-defined radio equipped with rapid modulation recognition," *IEEE Trans. Veh. Technol.*, vol. 59, no. 4, pp. 1659–1667, May 2010.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[22] Y. H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017.

[23] Y. He, F. R. Yu, N. Zhao, V. C. M. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 31–37, Dec. 2017.

[24] G. Gui, H. Huang, Y. Song, and H. Sari, "Deep learning for an effective non-orthogonal multiple access scheme," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8440–8450, Sep. 2018.

[25] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Machine learning for wireless networks with artificial intelligence: A tutorial on neural networks," unpublished paper, 2017. [Online]. Available: http://arxiv.org/abs/1710.02913

[26] D. Wang *et al.*, "Modulation format recognition and OSNR estimation using CNN-based deep learning," *IEEE Photon. Technol. Lett.*, vol. 29, no. 19, pp. 1667–1670, Oct. 2017.

[27] Y. Lcun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[28] T. J. OShea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *Proc. Int. Conf. Eng. Appl. Neural Netw.*, 2016, pp. 213–226.

[29] T. J. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[30] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 3, pp. 433–445, Sep. 2018.

[31] A. Ali, Y. Fan, and L. Shu, "Automatic modulation classification of digital modulation signals with stacked autoencoders," *Digit. Signal Process.*, vol. 71, pp. 108–116, 2017.

[32] P. E. Keller, *Artificial Neural Networks: An Introduction*. Bellingham, WA, USA: SPIE, 2005.

[33] V. Koltchinskii, "Neural network learning: Theoretical foundations," *Annales De L Institut Henri Poincaré Probabilités Et Statistiques*, vol. 39, no. 6, pp. 943–978, 2003.

[34] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[35] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.

[36] P. Koniusz, F. Yan, and K. Mikolajczyk, *Comparison of Mid-Level Feature Coding Approaches and Pooling Strategies in Visual Concept Detection*. Amsterdam, The Netherlands: Elsevier, 2013.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," unpublished paper, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[38] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

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

Fan Meng was born in Jiangsu, China, in 1992. He received the B.S. degree in 2015 from the School of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu, China, and is currently working toward the Ph.D. degree in the School of Information Science and Engineering, Southeast University, Nanjing, China. His main research topic is applying machine learning techniques in the wireless communication systems. His research interests include machine learning in general, joint demodulation and equalization, channel coding, resource allocation, end-to-end communication system design with machine learning tools.

Peng Chen (S'15–M'17) was born in Jiangsu, China, in 1989. He received the B.E. degree in 2011 and the Ph.D. degree in 2017, both from the School of Information Science and Engineering, Southeast University, Nanjing, China.

From March 2015 to April 2016, he was a Visiting Scholar with the Electrical Engineering Department, Columbia University, New York, NY, USA. He is currently an Associate Professor with the State Key Laboratory of Millimeter Waves, Southeast University. His research interests include radar signal processing and millimeter wave communication.

Lenan Wu received the M.S. degree in the electronics communication system from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1987, and the Ph.D. degree in signal and information processing from Southeast University, Nanjing, China, in 1997.

Since 1997, he has been a Professor with the Southeast University, and the Director of Multimedia Technical Research Institute. He is currently the author or co-author of more than 400 technical papers and 11 textbooks, and is the holder of 20 Chinese patents and one international patent. His research interests include multimedia information systems and communication signal processing.

Xianbin Wang (S'98–M'99–SM'06–F'17) received the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2001.

He is currently a Professor and a Tier-I Canada Research Chair with the Western University, London, ON, Canada. Prior to joining Western University, he was with Communications Research Centre Canada as a Research Scientist/Senior Research Scientist between July 2002 and December 2007. From Janaury 2001 to July 2002, he was a System Designer with STMicroelectronics, where he was responsible for the system design of DSL and Gigabit Ethernet chipsets. His current research interests include 5G technologies, Internet of Things, communications security, machine learning, and locationing technologies.

Dr. Wang has authored or co-authored more than 300 peer-reviewed journal and conference papers, in addition to 26 granted and pending patents and several standard contributions. He is currently a Fellow of Canadian Academy of Engineering, and an IEEE Distinguished Lecturer. He was the recipient of many awards and recognitions, including Canada Research Chair, CRC Presidents Excellence Award, Canadian Federal Government Public Service Award, Ontario Early Researcher Award, and five IEEE Best Paper Awards. He is currently an Editor/Associate Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON BROADCASTING, and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and He was also an Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS between 2007 and 2011, and the IEEE WIRELESS COMMUNICATIONS LETTERS between 2011 and 2016. He was involved in many IEEE conferences including GLOBECOM, ICC, VTC, PIMRC, WCNC, and CWIT, in different roles, such as Symposium Chair, Tutorial Instructor, Track Chair, Session Chair, and TPC Co-Chair.

Authorized licensed use limited to: HEFEI UNIVERSITY OF TECHNOLOGY. Downloaded on December 17,2021 at 07:38:38 UTC from IEEE Xplore. Restrictions apply.