This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2900460, IEEE Transactions on Vehicular Technology

Data-Driven Deep Learning for Automatic Modulation Recognition in Cognitive Radios

Yu Wang, *Student Member, IEEE*, Miao Liu, *Member, IEEE*, Jie Yang, *Member, IEEE*, and Guan Gui, *Senior Member, IEEE*

***Abstract*—Automatic modulation recognition (AMR) is an essential and challenging topic in the development of the cognitive radio (CR), and it is a cornerstone of CR adaptive modulation and demodulation capabilities to sense and learn environments and make corresponding adjustments. AMR is essentially a classification problem, and deep learning achieves outstanding performances in various classification tasks. So, this paper proposes a deep learning-based method, combined with two convolutional neural networks (CNNs) trained on different datasets, to achieve higher-accuracy AMR. A CNN is trained on samples composed of in-phase and quadrature component signals, otherwise known as in-phase and quadrature (IQ) samples, to distinguish modulation modes that are relatively easy to identify. We adopt dropout instead of pooling operation to achieve higher recognition accuracy. A CNN based on constellation diagrams is also designed to recognize modulation modes that are difficult to distinguish in the former CNN, such as 16QAM and 64QAM, demonstrating the ability to classify QAM signals even in scenarios with a low signal-to-noise ratio.**

***Index Terms*—Automatic modulation recognition (AMR), cognitive radio (CR), deep learning, convolutional neural network (CNN), in-phase and quadrature (IQ) samples, constellation diagrams.**

I. Introduction

A cognitive radio (CR) is a radio device capable of sensing, learning, and adjusting to adapt to external wireless environments [1], [2]. There are many types of modulation technologies, and one of the most essential functions in CR is to automatically select these modulation modes according to external environments. So, a precondition of receivers in demodulating received signals is to confirm signal modulation modes in CR; otherwise, the signals cannot be demodulated correctly, and transmission can't be completed. Therefore, automatic modulation recognition (AMR) is a function that must be solved in the CR receiver.

Due to state-of-the-art performance of deep learning in the physical layer [3]–[8], deep learning has been introduced gradually into CR for a range of tasks, most of which can be categorized as either classification, regression, or decision-making. For example, classification tasks consist of modulation recognition or wireless channel recognition, regression tasks involve modulation parameter estimation or estimation of primary user parameters, and decision-making is generally applied for channel assignment [9]–[12] or to select appropriate modulation modes to adapt to a wireless environment. AMR belongs to classification algorithms that can be categorized as either supervised or unsupervised. In this paper, we adopt supervised classification algorithms, which are trained on labeled data. For instance, a set of training data can be denoted as

$$T = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots (x_N, y_N)\}, \quad (1)$$

where $x_i \in X$ is the input variable; $y_i \in Y$ is the output variable or label, serving as a finite discrete variable in classification tasks; and $\forall i \in \{1, 2, \ldots, N\}$. The task of supervised classification algorithms is to identify a mapping relationship or function f from X to Y, which can minimize the following risk function,

$$R(f) = R_{emp}(f) + \lambda J(f), \quad (2)$$

where $R_{emp}(f)$ is an empirical risk function indicated as

$$R_{emp}(f) = \frac{1}{N}\sum_{i=1}^{N} L(x_i, f(x_i)). \quad (3)$$

The empirical risk function involves the loss of training data and can be used to express the risk function directly given sufficient training data. However, training data cannot meet requirements in reality, and when only using the empirical risk function with a small amount of data, over-fitting will occur. Over-fitting is a phenomenon where the trained model performs well in training data but performs poorly with test data; over-fitting occurs when the trained model is too complex. We generally use the structural risk function $R_{emp}(f) + J(f)$ in place of the empirical risk function $R_{emp}(f)$, in which $J(f)$ measures the complexity of the model and $\lambda \geq 0$ maintains a balance between the model complexity and empirical risk function.

II. Related works

AMR is defined as an algorithm that realizes modulation recognition of unknown signals without auxiliary information, depending solely on signals. AMR generally has three steps involving preprocessing of modulated signals, feature extraction, and classification of extracted features. Preprocessing can remove noise, estimate parameters of modulated signals, or transform signals into different forms to facilitate automatic extraction of features from neural networks. In a classical algorithm, wavelet transform and high-order cumulant extraction

This work was funded by the Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions, 1311 Talent Plan of NJUPT. *Corresponding author: Guan Gui*

The authors are with the Key Laboratory of Ministry of Education in Broadband Wireless Communication and Sensor Network Technology, Nanjing University of Posts and Telecommunications, Nanjing 21003, China (E-mail: {1018010407, liumiao, jyang, guiguan}@njupt.edu.cn)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2900460, IEEE Transactions on Vehicular Technology

are applied to extract features, and a decision tree or support vector machine relying on those features can classify a few modulation modes. In addition, neural networks have been gradually introduced into this field due to their remarkable performance in classification tasks. Paper [13] proposed that a convolutional neural network (CNN) trained on IQ samples could be applied to extract features automatically and effectively classify modulated signals; however, it is difficult to distinguish between 16QAM and 64QAM using this method.

III. Proposed methods

A. System description

Our system is combined with two CNNs and is designed for recognition of eight modulation modes of BPSK, QPSK, 8PSK, GFSK, CPFSK, PAM4, 16QAM, and 64QAM. These modulation modes are widely used in modern communication systems, including optical communications and satellite communications. When unknown signals are detected, the initial CNN trained on IQ samples is employed to recognize easily distinguishable modulation modes except 16QAM and 64QAM. This CNN does not have the capacity to distinguish between them, but it can separate them from other modulation modes. Therefore, they are categorized into the same class (QAMs), from which the other CNN trained on constellation diagrams can distinguish 16QAM and 64QAM.

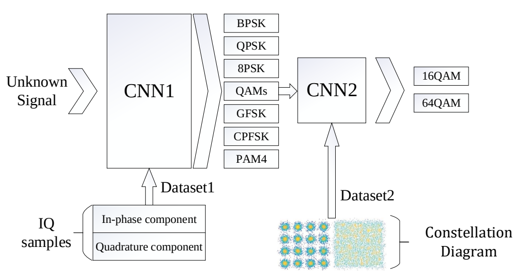

Fig. 1. Architecture of the system.

1) Former CNN: The former CNN has six layers (two convolution layers and four fully connected layers), structured as shown in Table I. The parametric rectified linear unit (PReLU) is selected as the activation function for all available layers except the last fully connected layer, where Softmax is applied to obtain the probability distribution matrix of the last layer. Assuming z_i as the independent variable and p_i as the other independent trainable variable, two activation functions are adopted as

$$f_{PReLU}(z_i) = \begin{cases} z_i, & if\ z_i > 0 \\ p_i z_i, & if\ z_i \le 0, \end{cases} \tag{4}$$

$$f_{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}. \tag{5}$$

In addition, cross entropy is introduced in this paper as a loss function to measure deviation between real values and predicted values. Assuming that $p(x)$ is the real distribution of data x and $q(x)$ is the predicted distribution, the cross entropy loss function is given as

$$\mathrm{H}(p,q) = -\sum_x p(x) \log q(x). \tag{6}$$

Different from the CNN proposed in [13], where the maximum pooling operation is applied, dropout (rather than a pooling operation) follows every convolution layer behind in the former CNN. Dropout replaces the pooling operation in this case because the pooling operation involves down-sampling, which can result in loss of signal characteristics, and dropout will not make signals neglect important features. Additionally, pooling is applied for dimensionality reduction to accelerate computation and avoid overfitting when the network size is too large; However, our network is not large, and dropout can avoid overfitting. The former CNN is denoted as DrCNN, the details of which are listed in Table I; the CNN in [13] is labeled as MaxCNN.

TABLE I
Layers of DrCNN and activation functions and output dimensions of every layer.

Layer	Output dimensions
Input	$1 \times 2 \times 128$
Conv2D (filters 128, size 2×8) + PReLU	$128 \times 1 \times 121$
Dropout (0.5)	/
Conv2D (filters 64, size 1×16) + PReLU	$64 \times 1 \times 114$
Dropout (0.5)	/
Flatten	7296
Dense + PReLU	128
Dropout (0.5)	/
Dense + PReLU	64
Dropout (0.5)	/
Dense + PReLU	32
Dropout (0.5)	/
Dense + Softmax	modulation modes

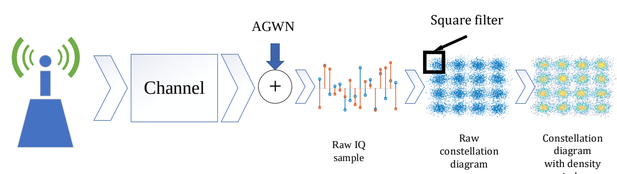

Fig. 2. Production process of dataset.

2) Latter CNN: The latter CNN is used to solve the classification problem between 16QAM and 64QAM through a constellation diagram with a density window; the structure is shown in Table II. Similar to the former CNN, Softmax is adopted in the last fully connected layer, and PReLU is chosen as an activation function for all convolution layers and other fully connected layers. In addition, this CNN applies the same loss function to calculate loss. However, this CNN has many differences from the former CNN, including more trainable parameters, adoption of a 3×3 or 5×5 convolution kernel rather than 1×16 or 2×8, and use of an average pooling operation because the training and test samples are constellation diagrams with local image correlation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2900460, IEEE Transactions on Vehicular Technology

TABLE II
LAYERS OF LATTER CNN AND ACTIVATION FUNCTIONS AND OUTPUT DIMENSIONS OF EVERY LAYER.

Layer	**Output dimensions**
Input	$128 \times 128 \times 3$
Conv2D(filters 128, size 5×5) + PReLU	$124 \times 124 \times 128$
Avepooling (size = 2, stride = 2)	$62 \times 62 \times 128$
Conv2D(filters 64, size 3×3) + PReLU	$62 \times 62 \times 64$
Conv2D(filters 64, size 3×3) + PReLU	$62 \times 62 \times 64$
Avepooling (size = 2, stride = 2)	$30 \times 30 \times 64$
Conv2D(filters 32, size 3×3) + PReLU	$30 \times 30 \times 32$
Conv2D(filters 32, size 3×3) + PReLU	$30 \times 30 \times 32$
Avepooling (size = 2, stride = 2)	$15 \times 15 \times 32$
Flatten	7200
Dense + PReLU	1024
Dropout (0.5)	/
Dense + PReLU	512
Dropout (0.5)	/
Dense + Softmax	modulation modes

B. Dataset

For the AMR task, we create two datasets of IQ samples and constellation diagrams with a density window. After raw signals are modulated by PSK, QAM, and others, they pass through a wireless channel in harsh communication environments, including additive Gaussian noise and multipath fading. We can obtain time-domain complex baseband signals as training or test data taken from 10 different signal-to-noise ratios (SNRs) ranging from -8 dB to 18 dB with an interval of 2 dB. Data for every SNR contains 6000 samples divided randomly into training samples and test samples by 7:3.

1) IQ samples: IQ samples are applied to train the former CNN. From time-domain complex baseband signals, a length-N complex vector is obtained as the training sample, denoted as

$$S = \left\{ s^{(1)}, s^{(2)}, s^{(3)}, \ldots, s^{(N)} \right\}, \tag{7}$$

where N represents the number of sampling points, and $s^{(i)}$, $\forall i \in \{1, 2, \ldots, N\}$ denotes a complex value of the signal-sampling point. Then, the complex vector is converted into two length-N real vectors: S_{Re} and S_{Im}, which are the real and imaginary parts of the complex vector, respectively. We then combine them into a $2 \times N$ matrix, denoted as

$$TS = \begin{pmatrix} S_{Re} \\ S_{Im} \end{pmatrix}, \tag{8}$$

which is fed into networks for training and testing. Because S_{Re} and S_{Im} also represent in-phase and quadrature components of signals, respectively, *TS* is also referred to as the IQ sample.

2) Constellation diagrams with density window: Constellation diagrams with a density window are adopted to train the latter CNN. Unlike the former network, the length-N complex vector *S*, where the value of N is generally great enough, is converted into a set of N points on the complex plane, denoted as

$$SNP = \left\{ \left(s_{Re}^{(1)}, s_{Im}^{(1)} \right), \left(s_{Re}^{(2)}, s_{Im}^{(2)} \right), \ldots, \left(s_{Re}^{(N)}, s_{Im}^{(N)} \right) \right\}, \tag{9}$$

the distribution of which is called a constellation diagram.

Then, a small square filter with stride 1 performs a convolution operation on the constellation diagram to calculate its data densities. Specifically, we apply a small square filter to slide on the constellation diagram with fixed-stride 1 to calculate the number of points in the field of the small square filter, labeled *d*. We denote the small square filter as a density window and call *d* the density. Because the points in constellation diagrams are not evenly distributed, the *d* calculated for every slide is different. We divide *d* into three degrees of higher density, moderate density, and lower density and apply yellow, green, and blue to represent this three degrees, respectively.

C. Implementation platform

All data production and preprocessing are accomplished using a GNU radio and MATLAB. GPUs composed of two NVIDIA GeForce GTX 1080Ti are applied for training and testing networks, and the implementation of networks is based on Keras with Tensorflow as the backend.

IV. EXPERINMENT RESULTS

A. Former CNN experiment

The experimental results of different networks in the test data are depicted in Fig. 3. When SNR exceeds 2 dB, the accuracy of DrCNN can reach up to more than 95% and demonstrates a nearly 3% improvement compared with MaxCNN when SNR exceeds 4 dB. In addition, RNN, DNN, and Inception are used for comparison, whose performances are far less than DrCNN. RNN and DNN are basic neural

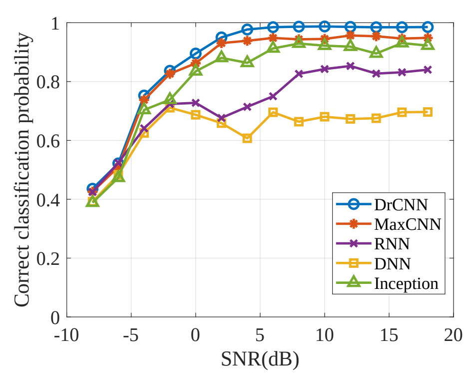

Fig. 3. Test accuracy of different networks in various SNRs.

networks; Inception, one species structure of CNN, is the major component of GoogLeNet designed by Google, which is pictured in Fig. 5. We adopt Inception for comparison to reveal that performance may not improve, even under conditions of increased parameters. Inception has 3,364,040 trainable parameters, far beyond DrCNN and MaxCNN, but does not perform as well as them. The parameters of five networks are listed in Table III.

Furthermore, we provide four confusion matrices of DrCNN in Fig. 4, which facilitate detailed analysis of our algorithm performance. From these confusion matrices, our algorithm can distinguish QAMs from other modulation modes with high accuracy, even with an SNR as low as -6 dB. Hence, our whole

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2019.2900460, IEEE Transactions on Vehicular Technology

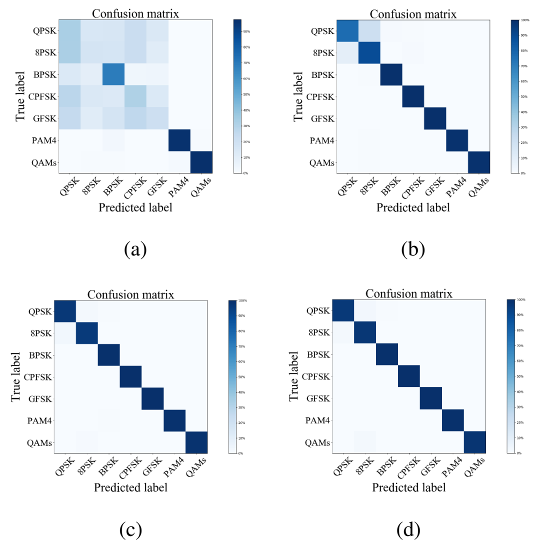

Fig. 4. Confusion matrix of DrCNN in various SNRs. With the sequences: (a) SNR = -6dB, (b) SNR = 0dB, (c) SNR = 6dB, (d) SNR = 12dB

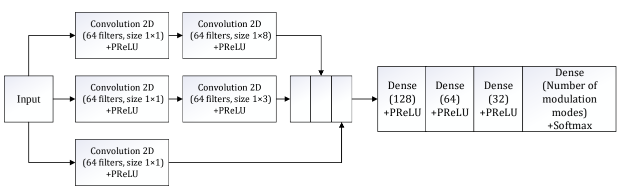

Fig. 5. Inception architecture.

system can run, because if the former CNN cannot effectively identify QAMs from all modulation modes in this paper, the latter CNN cannot work well, and our system is meaningless. Therefore, a precondition exists in which the former CNN can identify QAM signals for the latter CNN to classify QAM signals as either 16QAM or 64QAM.

TABLE III
PARAMETERS OF DIFFERENT NETWORKS.

Network	Parameters
DrCNN	1,101,448
MaxCNN	351,688
Inception	3,364,040
RNN	26,352
DNN	43,720

B. Latter CNN experiment

In the latter CNN experiment, we compare the performance of the CNN trained on constellation diagrams with DrCNN trained on $2 \times N$ IQ samples, the respective values of which are 128, 256, and 512; their performance is illustrated in Fig. 6. The CNN trained on constellation diagrams performs well with an accuracy of nearly 100%, whereas DrCNNs trained on $2 \times N$ IQ samples are incapable of distinguishing 16QAM and 64QAM.

V. CONCLUSION

In this correspondence, we have proposed a DL-based combination of two CNNs to identify different modulation

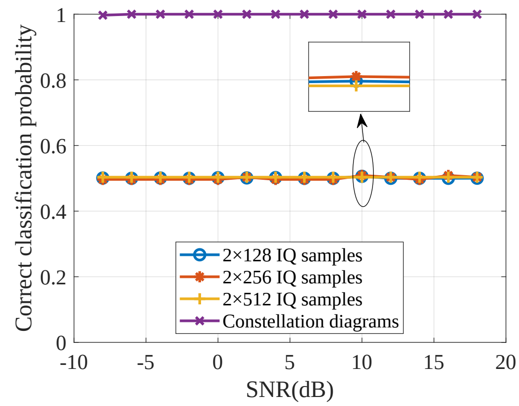

Fig. 6. Accuracy of CNN trained on constellation diagrams and DrCNN trained on IQ samples with different dimensions.

modes. Our proposed method can recognize eight modulation modes with high identification accuracy. Dropout is applied to replace the pooling operation to obtain better performance in the former CNN, and the latter CNN trained on constellation diagrams with a density window can detect sufficient differences to distinguish 16QAM and 64QAM, which cannot be differentiated in the former CNN. Due to the excellent performance of our method, we believe our approach can be directly employed in CRs if networks are trained by more signals with different modulation modes under different SNRs.

REFERENCES

[1] Y. C. Liang, et al.,, "Cognitive radio networking and communications: An overview," *IEEE Trans. Veh. Technol.*, vol. 60, no. 7, pp. 3386-3407, 2011.
[2] M. Liu, et al.,"Deep learning-inspired message passing algorithm for efficient resource allocation in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 641-653, Jan. 2019.
[3] W. Wang, et al., "Deep learning for wireless physical layer: Opportunities and challenges," *China Commun.*, vol. 14, no. 11, pp. 92-111, Nov. 2017.
[4] G. Gui, et al.,"Deep learning for an effective non-orthogonal multiple access scheme," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8440-8450, Sept. 2018.
[5] H. Huang, et al., "Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8549-8560, Sept. 2018.
[6] C. Lu, et al.,"MIMO channel information feedback using deep recurrent network," *IEEE Commun. Lett.*, vol. 23, no. 1, pp. 188-191, Jan. 2018.
[7] H. Huang, et al.,"Deep-learning-based millimeter-wave massive MIMO for hybrid precoding," *IEEE Trans. Veh. Technol.*, to be published, doi: 10.1109/TVT.2019.2893928
[8] Y. Tu, Y. Lin, J. Wang and J. Kim, "Semi-supervised learning with generative adversarial networks on digital signal modulation classification," *CMC-Comput. Mat. & Continua*, vol. 55, no. 2, pp. 243-254, May 2018.
[9] Z. Md. Fadlullah, et. al., "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Commun. Surveys and Tuts.*, vol. 19, no. 4, pp. 2432-2455, May 2017.
[10] N. Kato, et. al., "The deep learning vision for heterogeneous network traffic controlCproposal, challenges, and future perspective," *IEEE Wireless Commun. Mag.*, vol. 24, no. 3, pp. 146-153, Dec. 2016.
[11] F. Tang, et al.,"On a novel deep-learning-based intelligent partially overlapping channel assignment in SDN-IoT," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 80-86, Sept. 2018.
[12] B. Mao et al., "A novel non-supervised deep-learning-based network traffic control method for software defined wireless networks," *IEEE Wireless Commun.*, vol. 25, no. 4, pp. 74-81, Aug. 2018.
[13] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cognitive Commun. Net.*, vol. 3, no. 4, pp. 563-575, Dec. 2017.