IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 20, 2020, accepted April 3, 2020, date of publication April 20, 2020, date of current version May 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2988727

Cross Model Deep Learning Scheme for Automatic Modulation Classification

HONGBIN MA[1], GUANGYING XU[1], HUIXIAO MENG[1], MIN WANG[2], SHUYUAN YANG[1], (Senior Member, IEEE), RUOWU WU[3], AND WEI WANG[4]

[1]School of Artificial Intelligence, Xidian University, Xi'an 710071, China
[2]Key Laboratory of Radar Signal Processing, Xidian University, Xi'an 710071, China
[3]State Key Laboratory of Complex Electromagnetic Environment Effects on Electronics and Information System, Luoyang 471003, China
[4]Department of Arms Engineering, Academy of Armored Force Engineering, Beijing 100072, China

Corresponding author: Shuyuan Yang (syyang2009@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 91438103, and Grant 61771380, in part by the Foundation of the State Key Laboratory of CEMEE under 2018K0101B, and Grant 2019Z0101.

ABSTRACT Deep Neural Networks (DNNs) have achieved remarkable accuracy improvements for automatic modulation classification. However, the employed networks often have millions of parameters and need very high computation, which makes it difficult to deploy these models on portable devices with limited resources. We propose a cross model deep learning scheme to build a lightweight deep network for accurate modulation classification. Firstly, a large Hybrid DNN (HDNN) that is composed of convolutional and recurrent layers is constructed and trained for automatic and accurate classification of signals. Then we build a smaller Layered Resnet Network (LRN) with shallow layers and few nodes. The HDNN and LRN are taken as a Teacher Model (TM) and a Student Model (SM) respectively. Finally, a knowledge distillation method is proposed to guide the learning of the SM, by formulating a teaching loss from the prediction of the TM to train the SM. The performances of the proposed HDNN and LRN are investigated on the public RadioML2016.10a and RadioML2016.10b data sets. The experimental results show that the trained HDNN presents state-of-the-art classification results and the LRN trained in this scheme takes only about a sixth of the HDNN's inference time and consumes only 472.3KB for storage, with a slight accuracy decrease compared with the large HDNN.

INDEX TERMS Automatic modulation classification, cross model deep learning, layered Resnet network.

I. INTRODUCTION

Automatic modulation classification (AMC) aims to recognize the modulation type of a received radio signal, such as BPSK, PAM, MPSK, and QAM. AMC is a promising technology in the spectrum monitoring field [1] as it enables the supervision of interference signals by identifying the modulation format of the received signal. In addition, AMC is a key technique to demodulate the received signal in non-cooperative communication systems [2], [3]. Consequently, AMC is widely applied in many applications, both civil and military [4]–[6]. However, with the rapid development of wireless communication technology in recent years, the radio environment is becoming increasingly disordered, making AMC more difficult.

The associate editor coordinating the review of this manuscript and approving it for publication was Sunil Karamchandani.

Traditional AMC methods can be roughly divided into two groups: feature-based methods and likelihood-based methods [7]. The feature-based methods [8]–[10] seek for some handcraft features to distinguish different types of signals, such as higher-order moment [11], instantaneous frequency [7], instantaneous phase [11] and cyclic cumulant [11]. However, finding reliable features relies too much on manual selection, resulting in unstable results [7], [12]. In the likelihood-based methods, likelihood functions of different hypotheses are first calculated using received signals. Then the results are compared with a certain threshold to make final classification decisions [12]. Compared with feature-based methods, likelihood-based methods treat both noises and channel models which reflect the propagation characteristics of signals as prior information, nevertheless, channel models are usually unavailable in practice [12]. Consequently, likelihood-based methods cannot adapt to a

dynamic or unknown channel [7]. In addition, their computational complexity is very high [7].

Recently Deep Neural Networks (DNN) have been used for simultaneous feature learning and classification of modulated signals [12]–[17]. DNNs such as Convolutional Neural Networks (CNNs) [7], [12], [15]–[18] or Long Short-Term Memory Recurrent Neural Networks (LSTM-RNNs) [19], [20] improved the accuracy of AMC methods. For example, in [18], Wang *et al*. used the eye diagram of signals and Lenet-5 for AMC. In [21] and [22], CNNs were used to deal with the complex-valued raw signals, and the classification results on RML2016.10a data set [23] showed that CNNs can achieve higher accuracy than traditional expert feature engineering.

In [24], Rajendran *et al*. introduced a LSTM-RNN model for AMC and indicated that LSTM-RNN models outperform CNN models with oversampled received signals at small or medium scales. Later Swami and Sadler [11] designed a new LSTM-RNN model comprised of a LSTM-RNN layer and two Fully-Connected (FC) layers. It can achieve high accuracy for automatic classification of six types of digital modulation signals with varying noises. What's more, Sainath *et al.* [25] proved that CNN is good at reducing frequency variations and LSTM-RNN is good at temporal modeling. CNN and LSTM-RNN are complementary for sequential data processing. Using this work, West and O'Shea proposed a model comprised of inception modules and LSTM-RNN to identify the modulation types of signals, and the results showed remarkable accuracy improvements over both CNN models and LSTM-RNN models in [26]. In [27], a model composed of CNN, LSTM-RNN and Gated Recurrent Unit Recurrent Neural Network (GRU-RNN) achieved a state-of-the-art performance. In [28], Sharan *et al.* employed a CNN and LSTM-RNN model for AMC, and investigated the feasibility and effectiveness of deep learning algorithms for AMC. In existing models combined by CNN and LSTM-RNN, the original CNN structure [11], [28] and the inception structure [26] are utilized. However, these CNN structures are difficult to train [29] and there is the vanishing gradient in their training processes.

On the other hand, the available networks often have a large number of parameters, which makes it difficult to implement on portable devices with limited resources. For example, in [28] the network contains 313,603 parameters with one LSTM-RNN layer, four convolution layers, and two fully connected layers. In addition, the time complexity of the models comprised of CNN and LSTM-RNN layers is high in training or prediction as the LSTM-RNN operation is time-consuming [30]. Thus, these networks take a long time to automatically predict the types of signals.

Deep Residual nets (Resnet) [29] have been applied extensively in the field of computer vision. Resnet can simplify the training complexity of deep networks as the shortcut connection is adopted [29]. In order to limit the training complexity, in this paper, we utilize 1-D Resnet and LSTM-RNN layers to build a Hybrid Deep Neural Network (HDNN) for AMC. HDNN can present promising results on multiple AMC data sets. Moreover, in order to reduce the storage and computational cost of HDNN for real-time applications, we propose a Cross Model Deep Learning (CMDL) scheme to build a lightweight deep model for accurate prediction. We first construct a smaller Layered Resnet Network (LRN) with shallow layers and few nodes. Then, inspired by the Knowledge Distillation (KD) that builds a small and efficient model with reasonable performance degradation from a large and complex model, we define HDNN and LRN as a Teacher Model (TM) and a Student Model (SM) respectively. A Knowledge Distillation (KD) method is proposed to pilot the learning of the SM by formulating a teaching loss from the prediction of the TM to train the SM. In the training of the LRN, the inter-class similarity learned and revealed by the HDNN, is used to develop a more reliable LRN.

Compared with the available works, the contributions of our work can be summarized as follows:

- We propose a HDNN composed of Resnet and LSTM-RNN. We employ 1-D Resnet to reduce the training complexity of this model. To the best of our knowledge, this is the first attempt to combine 1-D Resnet and LSTM-RNN for AMC.
- In order to implement HDNN rapidly, we construct a lightweight deep model, LRN. This model is comprised of only three Resnet stacks and one FC layer.
- We propose a CMDL scheme to make the LRN achieve accurate prediction, where we utilize a KD method to guide the learning of the LRN, by formulating a teaching loss from the prediction of HDNN to train the LRN.

We analyze the performance of the trained LRN on the public RadioML2016.10a and RadioML2016.10b data sets. The experimental results show that the trained HDNN presents state-of-the-art classification results, and the trained LRN achieves a slight accuracy decrease compared with the HDNN and is beneficial to the rapid signal classification with limited resources. The trained LRN takes only about a sixth of the HDNN's inference time and occupies 472.3KB storage. In order to further compress LRN, the model is also quantized in experiments, the quantized LRN consumes only 301.2KB for storage, with the same performance or a slight accuracy increase compared with the trained LRN.

The remaining part of this paper is elaborated as follows. In Section II, CMDL is described in detail. Experimental results and related analysis are presented in Section III. Finally, a conclusion is drawn in Section IV.

II. CROSS MODEL DEEP LEARNING

In this section, a large HDNN and a small LRN are first constructed as TM and SM respectively. Then, the CMDL scheme is proposed to train LRN from HDNN.

A. CONSTRUCTIONS OF TM AND SM

In this subsection, a large HDNN is first constructed, which consists of three Resnet stacks, one LSTM-RNN layer, and one FC layer.

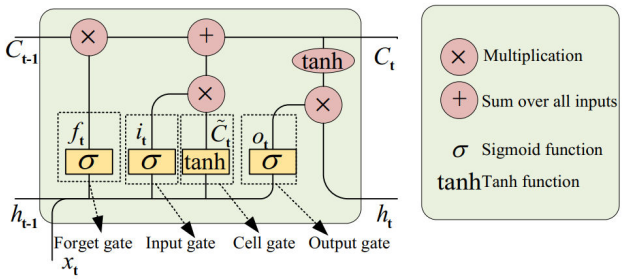


FIGURE 1. The structure of a LSTM-RNN cell. e_t is the input, C_t is the memory of this cell and h_t is its hidden state at time t.

A LSTM-RNN cell, which contains input gate (i_t), forget gate (f_t), output gate (o_t) and cell gate ($\tilde{C}_t$), is illustrated in Fig. 1. The gates are calculated as follows:

$$f_t = \sigma(\boldsymbol{W}_f \cdot [h_{t-1}, e_t] + b_f) \tag{1}$$
$$i_t = \sigma(\boldsymbol{W}_i \cdot [h_{t-1}, e_t] + b_i) \tag{2}$$
$$o_t = \sigma(\boldsymbol{W}_o \cdot [h_{t-1}, e_t] + b_o) \tag{3}$$
$$\tilde{C}_t = \tanh(\boldsymbol{W}_C \cdot [h_{t-1}, e_t] + b_C) \tag{4}$$

where f_t, i_t, o_t, $\tilde{C}_t$ are the forget gate, input gate, output gate and cell gate, respectively; $\boldsymbol{W}_f$, $\boldsymbol{W}_i$, $\boldsymbol{W}_o$ and $\boldsymbol{W}_C$ are forget gate, input gate, output gate and cell gate weight matrices, respectively; and b_f, b_i, b_o, b_C are forget gate, input gate, output gate and cell gate biases, respectively. e_t is the input at time t. The gate weights can be learned from the previous state h_{t-1} and the input data e_t.

The LSTM-RNN cell has the memory (C_t) and the hidden state (h_t) along with four gates. The memory C_t and the hidden state h_t at time t are updated as follows:

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{5}$$
$$h_t = o_t \cdot \tanh(C_t). \tag{6}$$

Utilizing this gating mechanism, LSTM-RNN cells can preserve information for a longer duration, thereby extracting temporal features.

In addition, 1-D convolution with low computation, as a complement to LSTM-RNN, also is employed to reduce frequency variations of signals for AMC. A 1-D convolution kernel can be described by

$$y_q[j] = g(\sum_{l=1}^{k} w_q[l]s[j+l-(k-1)/2]+b_q) \quad j \in [1, n] \tag{7}$$
$$y = [y_1, \ldots, y_q, \ldots, y_p] \quad 1 \le q \le p \tag{8}$$

where s denotes the input of the 1-D convolution kernel, and its length is n. Equation. (7) represents the operation of the $q-th$ 1-D convolution kernel in a 1-D convolution layer and k is the size of this convolution kernel where k is generally odd. $w_q[l]$ and b_q indicate the weights and the bias of this convolution kernel respectively. $g(\cdot)$ is an activate function and Glorot *et al.* [31] is used in our models. y is the output of the 1-D convolution layer and p is the number of its channel.

The number of the trainable parameters in a 1-D convolution kernel is $k+1$, and it only is about one $k-th$ of that in the 2-D convolution kernel with the size of $k \times k$ (The number of the trainable parameters in the 2-D convolution kernel is $k \times k + 1$).

Based on LSTM-RNN cells and 1-D convolution, the TM, a HDNN, is designed for AMC. As shown in Fig. 2 (a), a signal with the size of [2, 128] is fed into this model first, where the signal is a 128-sample complex (baseband I/Q) time-domain vector. Then we employ three ResNet stacks (Res1, Res1, and Res3), one LSTM-RNN layer (LSTM1) comprised of 100 LSTM cells, and one FC layer (FC1) to build the TM. The structure of the ResNet stack composed of 1-D convolution layers is presented in Fig. 2 (c). The standard cross-entropy loss is used as the loss for the training of the TM. It can be described as

$$L^T(x, l) = l \log(softmax(TM(x))) \tag{9}$$

where $x \in \mathrm{X}$, X is the training data set and l is the true label of x. $TM(x)$ denotes the output of the TM. $softmax(\cdot)$ is the softmax function.

In addition, as shown in Fig. 2, two ResNet stacks in the training TM, Res1 and Res2, are directly transferred to build the SM. Then, they are followed by one pooling layer (Pooling1) and one FC layer (FC2).

B. CROSS MODEL DEEP LEARNING

In fact, it is particularly difficult to make the SM obtain similar performance to the TM by using (9) as the SM, LRN, only has 84,939 parameters and it needs more parameters and more layers for higher accuracy.

In the CMDL scheme, we extend KD to train the SM from the TM, HDNN. We employ knowledge, $TM(x)$ and $F_{TM}(x)$, learned by the TM to improve the performance of the SM.

Based on KD, a loss where $TM(x)$ is introduced is designed to train the SM, and it can be described by

$$L_m^s(x, l) = softmax(\frac{TM(x)}{T}) \log(softmax(\frac{SM(x)}{T})) \tag{10}$$

where $SM(x)$ denotes the prediction of the SM. $L_m^s(x, l)$ is an inconsistency loss defined by the standard cross-entropy loss between $SM(x)$ and $TM(x)$, and it is utilized to make the SM learn from $TM(x)$.

Hinton *et al.* [32] proved that $softmax(\cdot/T)$ with a higher temperature value T produces a softer probability distribution over classes than that with a temperature value T of 1. The softer probability distribution makes the distillation pay more attention to matching the negative logits below the average, which is beneficial to train an efficient model [32]. With a high value T, the derivative w.r.t. $SM(x)$ is calculated in [32] by

$$\frac{\partial L_M^s(x, l)}{\partial SM(x)} \approx \frac{1}{num \times T^2}(SM(x) - TM(x)) \tag{11}$$

where num is the number of the modulation types. It is obvious that minimizing $L^s(x, l)$ can make $SM(x)$ equal to $TM(x)$ in (11). Thus, the SM is trained to approximate to the TM.

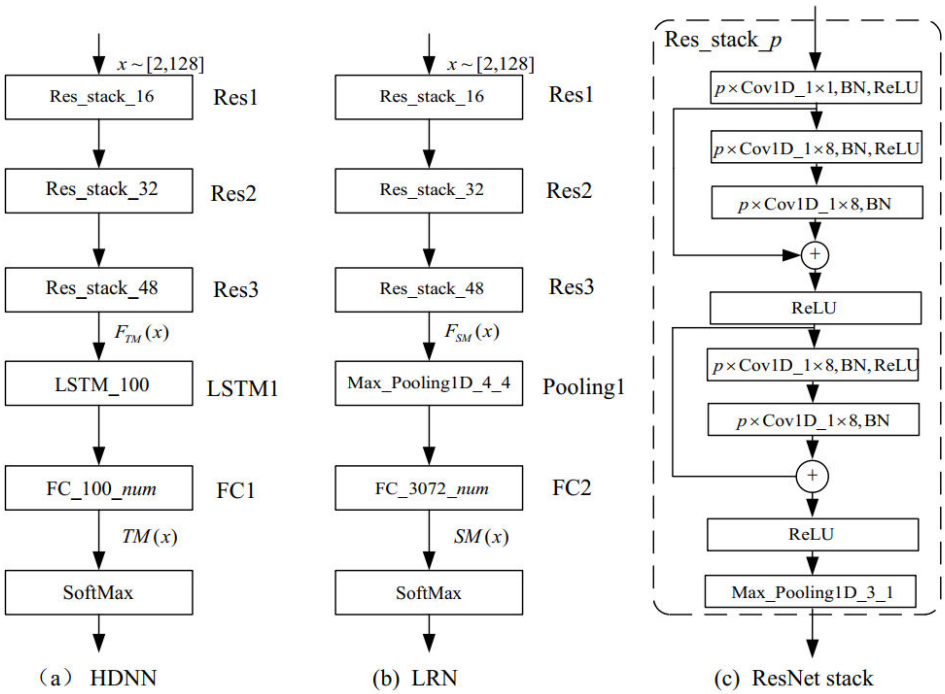


FIGURE 2. Structures of TM and SM. (a) The structure of the proposed HDNN. (b) The structure of the proposed LRN. (c) The structure of the ResNet stack. *p* × Cov1D_1 × k denotes a layer with *p* 1-D convolutional kernels, where the sizes of these convolutional kernels are 1 × k. Max_Pooling1D_s1_s2 is a 1-D max-pooling layer with the size s1 × 1 and the stride s2. FC_I_O is a Fully-connected layer with the input size of I and the output size of *O*. *x* is a signal as the input of models and its size is [2, 128]. *num* is the number of the modulation types to be identified. BN is a batch normalization layer. LSTM_100 denotes a LSTM-RNN layer comprised of 100 LSTM cells. In our scheme, HDNN is considered as the TM and LRN is the SM. HDNN is first trained. Then, we utilize *TM*(*x*) to guide the training of the SM.

In addition, in order to better learn knowledge from the TM, the SM is trained to learn the feature distribution of the TM by using (12) as features extracted by shallow layers are more generality [33].

$$L_f^s(x) = ||F_{TM}(x) - F_{SM}(x)||_2^2 \tag{12}$$

where $F_{TM}(x)$ and $F_{SM}(x)$ denotes the feature distribution of the TM and that of the SM respectively. The derivative w.r.t. $L_f^s(x, l)$ is:

$$\frac{\partial L_f^s(x)}{\partial F_{SM}(x)} = F_{TM}(x) - F_{SM}(x) \tag{13}$$

where we can find that minimizing $L_f^s(x)$ can make $F_{SM}(x)$ approximate to $F_{TM}(x)$ in (13).

In this paper, the teaching loss to train the SM is defined as

$$L = L_m^s(x, l) + L_f^s(x). \tag{14}$$

The CMDL is a multi-stage training process. Firstly, the TM is trained. Then, the SM is trained by knowledge generated by the trained TM. One important consideration in CMDL is summarized as follows.

- The construction and training of the TM are not necessary. Existing models with high accuracy for AMC can be considered as the TM.

III. SIMULATIONS, RESULTS, AND DISCUSSION

In this section, several experiments on RML2016.10a data set and RML2016.10b data set [23] are implemented to show the performance of the proposed CMDL.

The hardware of the test platform is HP z840 workstation with Intel E5-2600 3.2GHz CPU, 128G memory and two GTX 1080 GPU. All training processes are performed on two GPU and all testing processes are on one GPU. All experiments are implemented by Python 2.7 based on the Keras framework. The training log can be downloaded at this link, and all codes and data will be available.

In our experiments, all models are trained in an end-to-end manner by the Adam optimization algorithm in all experiments. The batch-size is set to 384. The learning rate is 0.01 and the learning rate decay rate is 0. In Adam optimization algorithm, exponential decay rates follow those provided in the original paper [34]. In addition, the training epoch is set to 300. T is set into 10.

A. DATASETS

The RML2016.10a data set generated with GNU Radio is adopted to evaluate the modulation recognition task, and it consists of 220,000 signals at SNRs of −20∼18 dB with 11 classes of modulations (8 digital and 3 analog types: 8PSK, AM-DSB, AM-SSB, BPSK, CPFSK, GFSK, PAM4,

TABLE 1. Comparisons of different models.

Model	Accuracy(%)	Training time(s/epoch)	Testing time(s/22000 signals)	Model size	The parameter number
TM*	62.95	142	320	648.9KB	128,743
SM*	55.20	30	54	472.3KB	84,939
SM(T=1)#	62.06	31	54	472.3KB	84,939
SM(T=3)#	62.18	31	54	472.3KB	84,939
SM(T=10)#	62.41	31	54	472.3KB	84,939
Quantized SM	62.41	—	57	301.2KB	84,939 (float16)

* denotes that this model is trained in the standard cross-entropy loss and # means that this model is trained in the teaching loss. Compared with the SM*, SMs trained in the teaching loss with T=1, T=3 and T=10 achieve the performance boost. When T is set to 10, the trained SM shows the highest accuracy. In addition the training time of the SM is only about a quarter of the TM's training time and the testing time of the SM is only about one-sixth of the TM's testing time.

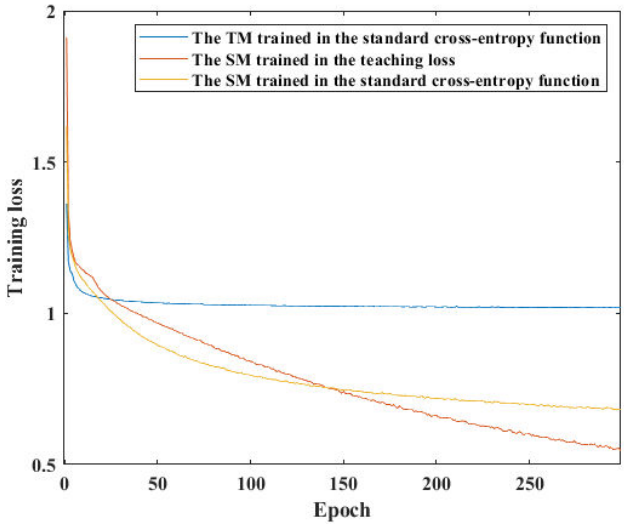


FIGURE 3. Variations of training losses.

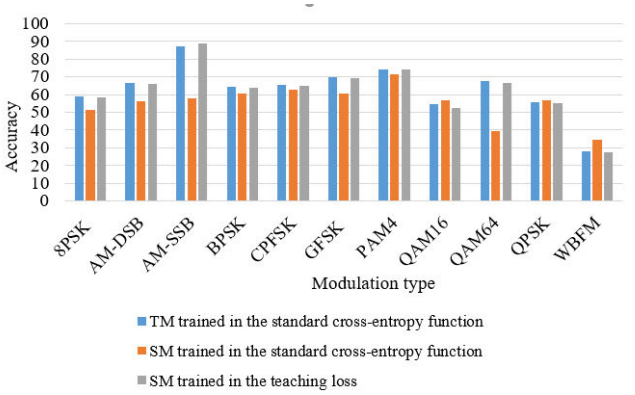


FIGURE 4. The accuracies of the proposed models in predicting for each class. The SM trained in the teaching loss shows a similar characteristic to that of the trained TM for different modulation types.

QAM16, QAM64, QPSK, and WBFM). The task is to utilize a signal represented by a 128-sample complex (baseband I/Q) time-domain vector to identify its modulation scheme out of 11 possible classes. The sample is fed into models in a 2×128 vector.

In order to better evaluate the proposed CMDL, a larger version of the RML2016.10a data set, the RML2016.10b data set, is also employed for AMC in our experiments. The RML2016.10b data set consists of 1,200,000 signals at SNRs of $-20\sim18$ dB with 10 classes of modulations (7 digital and 3 analog types: 8PSK, AM-DSB, BPSK, CPFSK, GFSK, PAM4, QAM16, QAM64, QPSK, and WBFM).

For ease of comparison, all data sets are download at https://www.deepsig.io/datasets. We utilize an official code downloaded from https://github.com/radioML/examples to split the data sets, where 90% of the data is considered as training data subset and 10% of the data is testing data subset in each data set.

B. THE TRAINING PROCESS

In this subsection, models are first trained. Then, we analyze the SM's performance variation with the temperature value. Next, we introduce the performance analysis of CMDL.

1) TRAINING OF MODELS

The proposed TM is first trained on the RML2016.10a training data subset in the standard cross-entropy loss. After the training in each epoch, the RML2016.10a testing data subset is employed to test the model. The model with the best accuracy on the testing data subset is saved as the trained model for prediction in the training process. As shown in Table 1, the TM achieves 62.98% accuracy on the testing data. Then, the proposed SM is trained on RML2016.10a training data subset in the standard cross-entropy loss, and it has an accuracy of 55.20% on the testing data. Next, the SM is trained on RML2016.10a training data subset in the proposed teaching loss, and the performance (62.41%) similar to that (62.98%) of the trained TM is obtained by the trained SM. Finally, in order to further compress the model, all parameters in the SM trained in the teaching loss are encoded by float16. The quantized SM obtains the same accuracy as the model without parameter quantization. The training loss variations of the TM and the SMs are illustrated in Fig. 3.

In order to show the SM's performance variation with the temperature value, when T is set to 1, 3, 10, the SM is trained on RML2016.10a training data subset, respectively. As illustrated in Table 1, prediction accuracy on RML2016.10a testing data subset increases with the increase of T. When T is 10, the SM achieves the best performance. Hence, for the remaining experiments, we will use a T of 10.

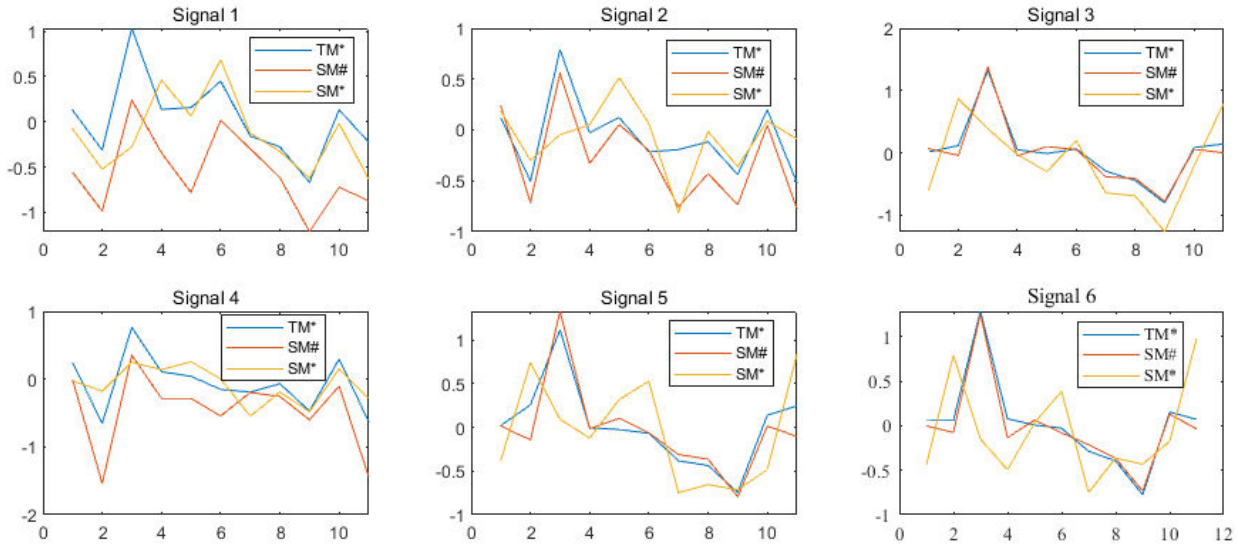


FIGURE 5. The visualization results of features from the last layers in SM and TM. * denotes that this model is trained in the standard cross-entropy loss and # means that this model is trained in the teaching loss. Compared with the feature difference between the SM* and the TM*, the feature difference between the SM # and the TM* is small, and their features change approximately in the same trend.

2) PERFORMANCE ANALYSIS OF CMDL

In fact, we expect the performance of the trained TM is better than that of the SM trained in the standard cross-entropy loss. As shown in Table 1, the trained TM exhibits significant performance improvement over the SM trained in the standard cross-entropy loss. Nevertheless, it is worth noting that the training time of the SM is only about a quarter of the TM's training time, and the testing time of the SM is only about one-sixth of the TM's testing time as the LSTM-RNN operation is very time-consuming [30]. What's more, as illustrated in Table 1, the teaching loss makes the SM obtain the performance improvement of 7.21%, which shows the proposed CMDL is effective for the performance boost of a lightweight model. One important reason is that the teaching loss employs knowledge learned by the trained TM to guide the training of the SM. However, an unexpected fact is that the quantized SM spent the same time as the SM without parameter quantization, which may be due to implementation issues. In theory, the quantized SM can cut computation time in half.

We utilize the trained models to predict all signals in the RML2016.10a testing data subset and plot the accuracies of the proposed models in predicting for each class. As shown in Fig. 4, the TM trained in the standard cross-entropy loss achieves the better performance than the SM trained in the standard cross-entropy loss for eight modulation types (8PSK, AM-DSB, AM-SSB, BPSK, CPFSK, GFSK, PAM4 and QAM64), and the SM trained in the standard cross-entropy loss shows the better performance for other modulation types (QAM16, QPSK and WBFM), which shows the characteristics of two models for the prediction of different modulation types. In addition, the SM trained in the teaching loss also obtains the better performance than the SM

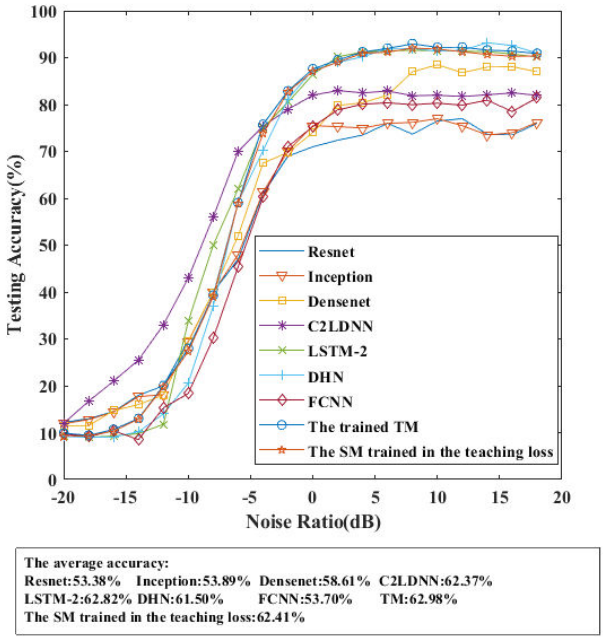


FIGURE 6. Comparisons with different methods. Compared with Resnet, Inception, Densenet, FCNN, DHN, C2LDNN and LSTM-2, the trained TM achieves much higher accuracy for AMC and the proposed SM trained in the teaching loss shows a better performance.

trained in the standard cross-entropy loss for eight modulation types (8PSK, AM-DSB, AM-SSB, BPSK, CPFSK, GFSK, PAM4 and QAM64) and its characteristic for the prediction of different modulation types is changed compared with the SM trained in the standard cross-entropy loss, which is similar to the characteristics of the trained TM. This proves that knowledge obtained by the TM is learned effectively by the SM using the proposed teaching loss.

In addition, when we use the trained models to test randomly selected signals in the RML2016.10a testing data subset, features from the last layers in the trained SMs and the trained TM are visualized in Fig. 5. The feature difference between the SM trained in the teaching loss and the trained TM is small, and their features change approximately in the same trend, which further proves that knowledge obtained by the TM is effectively learned by the SM in the teaching loss and the features extracted by the HDNN is beneficial to performance improvement of the LRN for AMC.

C. THE EVALUATION OF THE PROPOSED MODELS

In this subsection, in order to further show the performance of the proposed models, seven state-of-the-art CNN and LSTM-RNN models (Resnet [26], Inception [26], Densenet [26], FCNN [7], DHN [35], C2LDNN [26] and LSTM-2 [24]) are introduced to quantify the experimental results on the RML2016.10a test data subset. Resnet, Inception and Densenet were used in [26] for AMC. FCNN [7], DHN [35], C2LDNN [26] and LSTM-2 [24] were proposed for AMC. These models are evaluated on the RML2016.10a data set in the original papers.

The experimental results are shown in Fig. 6. It can be noticed that the trained TM and the SM trained in the teaching loss achieve much higher accuracy than Resnet, Inception, Densenet, FCNN, and C2LDNN on testing signals at SNRs of $-5\sim18$ dB and they result in similar performance compared with other methods.

In view of the average accuracy, the best performance is illustrated by the trained TM, and either superior or equal performance is shown by the SM trained in the teaching loss.

D. THE GENERALITY OF THE PROPOSED CMDL

In this subsection, A Novel TM (NTM) and a Novel SM (NSM) are designed to show the generality of the proposed CMDL scheme on the RML2016.10b data set. The structures of the NTM and the NSM are shown in Fig. 7.

In order to further evaluate the performance of models, the NTM and the NSM are first trained in the standard cross-entropy loss. Then, the NSM is trained in the teaching loss. Finally, the trained NSM is quantized. The experimental results are shown in Table 2. Compared with the NSM trained in the standard cross-entropy loss, the NSM trained in the teaching loss has a performance boost of 5.8%, which illustrates the feasibility and the generality of the proposed CMDL

TABLE 2. Comparisons of different models.

Model	Accuracy	Training time	Testing Time	Model size
NTM*	63.78%	275s	29.4s	1396KB
NSM*	56.26%	50s	7.9s	1218KB
NSM(T=10)#	62.06%	50s	7.9s	1218KB
Quantized NSM	62.11%	—	7.9s	671.4KB

* denotes that this model is trained in the standard cross-entropy loss and # means that this model is trained in the teaching loss. Compared with the NSM*, the NSM# achieves the performance boost.

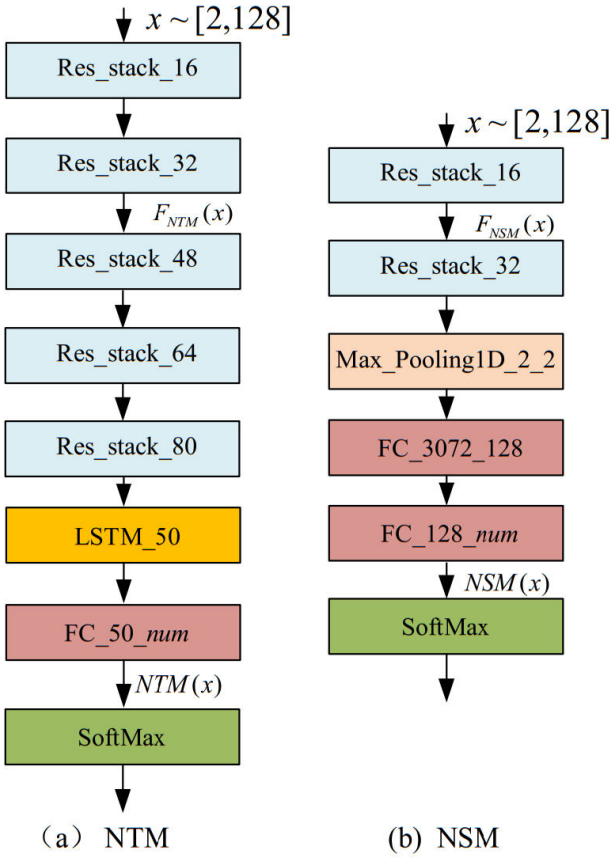


FIGURE 7. Structures of NTM and NSM. *NSM*(*x*) denotes the prediction of the NSM and *NTM*(*x*) denotes the prediction of the NTM.

on this model. The trained NSM with reasonable performance degradation from the trained NTM also saves a lot of time in the training process and the testing process. In addition, the quantized NSM has a slightly better performance than the original NSM trained in the teaching loss, which is probably because parameter quantization improves the generalization performance of this model.

IV. CONCLUSION AND FUTURE WORK

In this paper, in order to better train a lightweight model for AMC, we propose a novel scheme, CMDL. Firstly, we construct a large HDNN for AMC and this model achieves state-of-the-art performance compared with its counterparts. Then a lightweight model, LRN, is built. Next, a KD method is proposed by formulating a teaching loss from the prediction of the HDNN to train the LRN. The trained LRN that consumes only 472.3KB for storage achieves great performance improvement compared with the LRN trained in the standard cross-entropy loss, and results in either superior or equal performance compared with its counterparts, which proves that the proposed CMDL scheme is beneficial to train a lightweight model. In order to further compress the lightweight LRN, in experiments, parameter quantization is employed and the model size is reduced to 301.2KB with either higher or equal accuracy.

REFERENCES

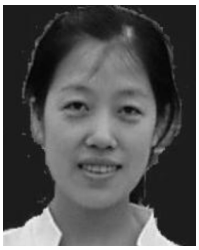
[1] S. Huang, Y. Jiang, X. Qin, Y. Gao, Z. Feng, and P. Zhang, "Automatic modulation classification of overlapped sources using multi-gene genetic programming with structural risk minimization principle," *IEEE Access*, vol. 6, pp. 48827–48839, 2018.
[2] S. I. H. Shah, S. Alam, S. A. Ghauri, A. Hussain, and F. A. Ansari, "A novel hybrid cuckoo search- extreme learning machine approach for modulation classification," *IEEE Access*, vol. 7, pp. 90525–90537, 2019.
[3] K. Zhang, E. L. Xu, Z. Feng, and P. Zhang, "A dictionary learning based automatic modulation classification method," *IEEE Access*, vol. 6, pp. 5607–5617, 2018.
[4] Z. Zhu and A. K. Nandi, *Automatic Modulation Classification: Principles, Algorithms and Applications*, 1st ed. New York, NY, USA: Wiley, 2015.
[5] 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.
[6] T. Ulversoy, "Software defined radio: Challenges and opportunities," *IEEE Commun. Surveys Tuts.*, vol. 12, no. 4, pp. 531–550, Nov. 2010.
[7] S. Zheng, P. Qi, S. Chen, and X. Yang, "Fusion methods for CNN-based automatic modulation classification," *IEEE Access*, vol. 7, pp. 66496–66504, 2019.
[8] D. Grimaldi, S. Rapuano, and L. De Vito, "An automatic digital modulation classifier for measurement on telecommunication networks," *IEEE Trans. Instrum. Meas.*, vol. 56, no. 5, pp. 1711–1720, Oct. 2007.
[9] S. Majhi, R. Gupta, W. Xiang, and S. Glisic, "Hierarchical hypothesis and feature-based blind modulation classification for linearly modulated signals," *IEEE Trans. Veh. Technol.*, vol. 66, no. 12, pp. 11057–11069, Dec. 2017.
[10] Z. Wu, S. Zhou, Z. Yin, B. Ma, and Z. Yang, "Robust automatic modulation classification under varying noise conditions," *IEEE Access*, vol. 5, pp. 19733–19741, 2017.
[11] A. Swami and B. M. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Trans. Commun.*, vol. 48, no. 3, pp. 416–429, Mar. 2000.
[12] S. Huang, L. Chai, Z. Li, D. Zhang, Y. Yao, Y. Zhang, and Z. Feng, "Automatic modulation classification using compressive convolutional neural network," *IEEE Access*, vol. 7, pp. 79636–79643, 2019.
[13] A. Ali and F. Yangyu, "Automatic modulation classification using deep learning based on sparse autoencoders with nonnegativity constraints," *IEEE Signal Process. Lett.*, vol. 24, no. 11, pp. 1626–1630, Nov. 2017.
[14] Y. Liu, O. Simeone, A. M. Haimovich, and W. Su, "Modulation classification via Gibbs sampling based on a latent Dirichlet Bayesian network," *IEEE Signal Process. Lett.*, vol. 21, no. 9, pp. 1135–1139, Sep. 2014.
[15] J. H. Lee, K.-Y. Kim, and Y. Shin, "Feature image-based automatic modulation classification method using CNN algorithm," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2019, pp. 1–4.
[16] S. Peng, H. Jiang, H. Wang, H. Alwageed, Y. Zhou, M. M. Sebdani, and Y.-D. Yao, "Modulation classification based on signal constellation diagrams and deep learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 3, pp. 718–727, Mar. 2019.
[17] F. Meng, P. Chen, L. Wu, and X. Wang, "Automatic modulation classification: A deep learning enabled approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10760–10772, Nov. 2018.
[18] D. Wang, M. Zhang, Z. Li, J. Li, M. Fu, Y. Cui, and X. Chen, "Modulation format recognition and OSNR estimation using CNN-based deep learning," *IEEE Photon. Technol. Lett.*, vol. 29, no. 19, pp. 1667–1670, Oct. 1, 2017.
[19] N. Daldal, Ö. Yıldırım, and K. Polat, "Deep long short-term memory networks-based automatic recognition of six different digital modulation types under varying noise conditions," *Neural Comput. Appl.*, vol. 31, no. 6, pp. 1967–1981, Jun. 2019.
[20] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," 2017, *arXiv:1710.09282*. [Online]. Available: http://arxiv.org/abs/1710.09282
[21] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *Proc. Int. Conf. Eng. Appl. Neural Netw.*, 2016, pp. 213–226.
[22] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cognit. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.
[23] T. J. O'Shea and N. West, "Radio machine learning dataset generation with GNU radio," in *Proc. GNU Radio Conf.*, Sep. 2016, pp. 1–6.
[24] 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. Cognit. Commun. Netw.*, vol. 4, no. 3, pp. 433–445, Sep. 2018.
[25] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 4580–4584.
[26] N. E. West and T. O'Shea, "Deep architectures for modulationrecognition," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Mar. 2017, pp. 1–6.
[27] S. Yao, Y. Zhao, H. Shao, S. Liu, D. Liu, L. Su, and T. Abdelzaher, "FastDeepIoT: Towards understanding and optimizing neural network execution time on mobile and embedded devices," in *Proc. 16th ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Shenzhen, China, 2018, pp. 278–291.
[28] S. Ramjee, S. Ju, D. Yang, X. Liu, A. El Gamal, and Y. C. Eldar, "Fast deep learning for automatic modulation classification," 2019, *arXiv:1901.05850*. [Online]. Available: http://arxiv.org/abs/1901.05850
[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[30] Y. Zhang, C. Wang, L. Gong, Y. Lu, F. Sun, C. Xu, X. Li, and X. Zhou, "A power-efficient accelerator based on FPGAs for LSTM network," in *Proc. IEEE Int. Conf. Cluster Comput. (CLUSTER)*, Sep. 2017, pp. 2168–9253.
[31] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.
[32] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 1–9.
[33] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. NIPS*, 2014, pp. 3320–3328.
[34] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–13.
[35] J. Nie, Y. Zhang, Z. He, S. Chen, S. Gong, and W. Zhang, "Deep hierarchical network for automatic modulation classification," *IEEE Access*, vol. 7, pp. 94604–94613, 2019.

HONGBIN MA received the B.S. degree in electronic and information engineering from the Shandong University of Technology, Zibo, China, in 2016. He is currently pursuing the Ph.D. degree in intelligent information processing with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, Xidian University, Xi'an, China.

GUANGYING XU received the B.S. degree in automation specialty from the China University of Petroleum, in 2016. He is currently pursuing the Ph.D. degree with the Institute of Intelligent Information Processing. Xidian University, Xi'an, China. His research interests include few-shot learning and hyperspectral image classification.

HUIXIAO MENG received the B.S. degree in electronic information engineering from the Hebei University of Science and Technology, Shijiazhuang, China, in 2007, and the M.S. degree in circuits and systems from Xidian University, Xi'an, China, in 2010, where she is currently pursuing the Ph.D. degree with the School of Artificial Intelligence. Her research interests include explainable deep learning and few-shot learning.

MIN WANG received the B.S. degree in automatic control and the M.S. and Ph.D. degrees in signal and information from Xidian University, Xi'an, China, in 2000, 2003, and 2005, respectively. He is currently an Associate Professor with the National Laboratory of Radar Signal Processing, Xidian University. His research interests include radar signal processing, statistical signal processing, and impulse radio.



SHUYUAN YANG (Senior Member, IEEE) received the B.S. degree in electrical engineering and the M.S. and Ph.D. degrees in circuit and system from Xidian University, Xi'an, China, in 2000, 2003, and 2005, respectively. She is currently a Professor with the Key Laboratory of Intelligent Perception and Image Understanding, Ministry of Education, Xidian University. Her research interests include intelligent signal processing, machine learning, and image processing.

RUOWU WU received the master's degree from the University of Electronic Science and Technology of China, in 2013. His main research interest includes characteristics and simulation of complex electromagnetic environments.

WEI WANG received the B.S. degree from the Department of Arms Engineering, Beijing, China. He is currently with the Department of Arms Engineering.