IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 25, 2019, accepted March 15, 2019, date of publication March 21, 2019, date of current version April 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2906564

A Gradually Distilled CNN for SAR Target Recognition

RUI MIN, (Member, IEEE), HAI LAN, (Student Member, IEEE), ZONGJIE CAO, (Member, IEEE), AND ZONGYONG CUI, (Member, IEEE)
School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

Corresponding author: Zongjie Cao (zjcao@uestc.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61801098, and in part by the Fundamental Research Funds for the Central Universities under Grant 2672018ZYGX2018J013.

ABSTRACT Convolutional neural networks (CNNs) have been widely used in synthetic aperture radar (SAR) target recognition. Traditional CNNs suffer from expensive computation and high memory consumption, impeding their deployment in real-time recognition systems of SAR sensors, as these systems have low memory resources and low speed of calculation. In this paper, a micro CNN (MCNN) for real-time SAR recognition system is proposed. The proposed MCNN has only two layers, and it is compressed from a deep convolutional neural network (DCNN) with 18 layers by a novel knowledge distillation algorithm called gradual distillation. MCNN is a ternary network, and all its weights are either -1 or 1 or 0. Following a student-teacher paradigm, the DCNN is the teacher network and MCNN is its student network. The gradual distillation makes MCNN a better learning route than traditional knowledge distillation. The experiments on the MSTAR dataset show that the proposed MCNN can obtain a high recognition rate which is almost the same as the DCNN. However, compared with the DCNN, the memory footprint of the proposed MCNN is compressed 177 times, and the calculated amount is 12.8 times less, which means that the proposed MCNN can obtain better performance with the smaller network.

INDEX TERMS SAR target recognition, micro convolutional neural network, knowledge distillation, model compression, ternary network.

I. INTRODUCTION

Synthetic Aperture Radar (SAR) is an active microwave imaging sensor. It utilizes the synthetic aperture principle to achieve high resolution microwave imaging. It has the advantages of all-day, all-weather, high resolution, and wide width [1], [2]. With these important benefits, SAR image classification has always been a research hotspot. However, since SAR images do not provide rich color information like optical images and there is a large amount of speckle noise in SAR image, SAR image classification is more difficult than optical image classification.

Deep learning algorithms such as deep convolutional networks have revolutionized the field of computer vision and have surpassed traditional image processing algorithms in many applications, especially optical image classification. Inspired by CNN's application in optical images, deep convolutional neural networks (CNNs) have been playing an important role in SAR target recognition over the past several years. Based on a large amount of data and parameters, CNNs can achieve high accuracy in SAR image classification [3]–[6].

Although CNN performs well in SAR image recognition, their parameters are increasing along with the computation required [7]–[9]. Therefore, despite the good performance, these traditional CNNs are difficult to deploy on embedded platform. Limited by small memory capacity and slow computational speed, it is hard for real-time recognition systems of SAR sensors to deploy these over-parameterized CNNs. Therefore, designing a small, fast, and accurate CNN is a core issue in deploying artificial intelligence in real-time recognition systems of SAR sensors.

Recently, there are two main ways to design small CNNs: designing structural matrix and model compressing. Designing structural matrix aims at using specially structured neural kernels to construct the network. Model compression

The associate editor coordinating the review of this manuscript and approving it for publication was Gerardo Di Martino.

obtains a small network by compressing existing mature networks.

In CNNs, traditional convolution kernels are two-dimensional matrices. Storing a convolution kernel whose size is $M \times N$ requires MN parameters. Designing structural matrix aims designing a special convolution kernel with less than MN parameters. This will reduce the network memory needed and also accelerate the computation. Works in [10] proposes SqueezeNet. Each layer in SqueezeNet contains multiple parallel convolution kernels with two types: one is 1×1 and the other is 3×3. Comparing with traditional CNNs, squeezenet can accomplish image recognition tasks with fewer parameters and faster speed because of its special structural design. Google proposed MobileNet in [11]. There are three different convolutional cores in MobileNet: One is the traditional convolutional kernel whose size is 1×1, one is the traditional convolutional kernel whose size is 3×3, and the other one is a deepwise convolutional kernal whose size is 3×3. In MobileNet, through ingenious combinations of the these three convolution kernels, the convolution kernel whose size is 1×1 occupies most of the memory and computational resources, which makes the model size of the network very small and computationally fast.

SqueezeNet and MobileNet are lightweight network models. They are all parallel network structures, that is, there are many branches in a network. With the hardware support of GPU and CPU, these branches can coordinate the operation and get the results. However, in SAR front-end or embedded devices, these conditions will not be met, which will lead to the inconsistency of branch computing in SqueezeNet and MobileNet, and then affect its operation speed. The solution to this problem is to design special scheduling algorithms or add hardware. As a result, they are not suitable for deployment in existing real-time radar sensor system.

Recent works on model compression can be classified into three main categories: parameter pruning and sharing; low-rank factorization; and knowledge distillation [12].

Parameter pruning removes unimportant weights and connections in the network to compress and accelerate the network. An early pruning algorithm removed weights whose values were small than a set threshold. The Optimal Brain Damage [13] and the Optimal Brain Surgeon in [14] use the Hessian matrix of the loss function to reduce the number of weights and connections. The above pruning algorithms are irreversible, important weights and connections may be pruned wrongly. Dynamic Network Surger in [15] proposed a dynamic pruning algorithm to recover those weights that are pruned in error. Dynamic Network Surger greatly reduces the parameters and connections in the CNNs.

Low-rank factorization decomposes traditional convolution kernels by some specific mathematical methods to make them sparse, which was done layer by layer. For example, the work in [16] attempts to replace the 2d convolution kernel with a separable 1d convolution kernel, following the dictionary leaning algorithm. For some convolutional kernels, some low-rank approximation and clustering schemes are proposed in [17]. They accelerate the network 2 times with only 1% loss in accuracy.

Network quantization is an important method in parameter sharing. Network quantization compresses the original network by reducing the number of bits required to represent each weight [12]. Traditional weights in CNNs are float represented by 32 bit, which is considered redundant. Works in [18] show that 8-bit quantization of the parameters is enough for some CNNs, and it can accelerate CNNs with little loss of accuracy. Based on CNN training, the work in [19] used 16-bit fixed-point representation with stochastic rounding to replace original weights, which significantly reduces model size and amount of calculation. More extreme, works in [20] proposed BinaryNet and works in [21] proposed XNOR-Net both usie 1-bit weights to build CNNs. In this extreme way, the memory footprint is greatly reduced, but it leads to a loss of precision that cannot be ignored.

As we know, knowledge distillation (KD) was first proposed in [22]. In their work, small networks are trained with the supervision of large trained networks. They aim to shift knowledge form big networks into small networks. In the process of knowledge distillation, those big networks are called teacher networks, and those small networks are called student networks. Through this operation, student networks can mime the function learned by the teacher networks. The knowledge we want to extract from the teacher network can be the teacher network's final probability distribution, and it also can be the dark knowledge defined in [23]. The work in [24] introduces a KD compression framework, which follows the student-teacher paradigm, in which students are punished by softened versions and real labels output by teachers. The framework compresses an ensemble of deep networks (teachers) into similar deep student networks. To this end, students were trained to predict teachers' output and real classification labels. Despite its simplicity, KD has outstanding performance in various classification tasks [25], [26].

Parameter pruning and low-rank factorization both require special algorithms and hardware to deploy on the embedded platform, so they are not suitable for existing real-time radar sensor systems. Parameter sharing and knowledge distillation do not require special hardware support, and models compressed by these two methods can be deployed in the existing real-time radar sensor system.

In this paper, a micro convolutional neural network (MCNN) and its training algorithm is proposed. MCNN is compressed from the deep convolutional neural network (DCNN) in [27]. To the best of our knowledge, this DCNN is the best performer on the MSTAR dataset without any data enhancement. MCNN has only two layers, and its all weights are either -1 or 1 or 0. Our compression algorithm is based on a combination of knowledge distillation and weight quantization. A novel training approach called gradual distillation is proposed to train MCNN. The novel approach is based on a traditional knowledge distillation algorithm and extends the idea for shallow quantized network. We validate the MCNN on MSTAR dataset and provide evidence that

our MCNN matches or outperforms the DCNN. Compared with DCNN, MCNN is faster and smaller enough to deploy on real-time SAR sensor system.

The overall structure of this paper is arranged as follows: in Section II, we briefly review the related works on SAR target recognition and model miniaturization. Section III presents the proposed MCNN and its training algorithm. In section IV, we compare the proposed MCNN with some deep CNNs and compare the proposed training algorithm with different KD training algorithms. Finally, Section V shows the conclusion.

II. RELATED WORK

A. SAR TARGET RECOGNITION VIA CNNS

SAR target recognition usually faces two challenges:one is deficient color information of SAR image, and the other is interference of speckle noise. The pixels of a SAR image have only gray information. Two objects with similar shape can be easily misclassified into the same category, which is particularly common when the resolution of SAR is low. Speckle noise can blur the shape of the target, making image recognition more difficult. Because of the above two difficulties, traditional methods perform poorly in SAR target classification. With the development of deep neural networks, a revolutionary breakthrough has been made in optical image classification. Inspired by this, many researchers apply deep learning to classify SAR target. In these studies, CNNs are used the most. Given real labels, CNNs automatically learn the useful information in SAR images to classify them. The influence of the speckle is small during the learning process. Sometimes in order to improve accuracy, researchers even artificially add noise. These all indicate that CNN has strong anti-interference ability and learning ability. It is feasible to classify SAR images with CNN.

A CNN with 4 layers is proposed in [28] to classify SAR images, and it achieves a recognition rate of 94.29% on the MSTAR dataset. Its performance exceeds most traditional machine learning methods. With the development of CNNs, more advanced network structures have been proposed to identify SAR targets. In [27], a CNN with 18 layers is proposed to classify MSTAR dataset, and it achieves the recognition rate of 98.8% without any data enhancement. This CNN network is based on ResNet [29]. With the help of Long Short-Term Memory and multi-branch architecture, works in [8] has not only increased the recognition rate of MSTAR to 99.9%, but also show the good antinoise and anti-confusion performance. High accuracy establishes the leading position of CNNs in the field of SAR image recognition. Based on this, CNNs are also used in other fields related to SAR images, such as target detection.

In order to reduce the model size of the CNN for identifying SAR targets, works in [30] propose a lightweight network. This lightweight network is compressed from a DCNN with 6 layers via network pruning, quantization and Huffman coding. They compress the weight storage by 40 times and reduce the number of multiplication by 15 times without loss of accuracy.

B. MODEL MINIATURIZATION

As described in the introduction, recent works on model compression can be classified into three main categories: parameter pruning and sharing; low-rank factorization; and knowledge distillation. These algorithms can be used individually or jointly. In order to compress the network greatly and retain its accuracy, these algorithms are used in combination.

Network pruning, quantization and Huffman encoding are used in [31] to deeply compress CNN, and this method is called deep compression. Deep compression removes the redundant connections via network pruning and quantize the weights, then use Huffman coding to code the quantized weights. Deep compression can reduce the model size by tens of times with little bit loss of accuracy. The quantization bits in deep compression is no less than 4 bit, and there is still room for reduction.

To further compress the model size, 1-bit and 2-bit quantization are proposed. Binary Network in [20] use -1 and 1 as network weights. It only takes 1 bit to store a weight in Binary Network. When the network is binarized, convolution or matrix multiplication can be simplified to bitwise XNOR and bit-count operations. Binaryzation directly compresses the model size by 32 times compared to a 32-bit network. However, a negligible decline of accuracy appears in binary networks. For this reason, ABCNet is proposed in [32]. In ABCNet, a set of linear combinations of binary filter base are used to represent network layer parameters, which extends the representation range of weights. Compared with BinaryNet, ABCNet has higher accuracy which is similar to 32 bit network. These binary networks use binary weights in forward propagation and 32 bit weights in backward propagation. Compared with 32 bit network, the training time of network has not decreased. DoReFa-Net is proposed in [33] for reducing the training time of quantized networks. DoReFa-Net is a method to train quantized CNNs. Especially, during backward pass, parameter gradients are stochastically quantized to low bitwidth numbers before being propagated to convolutional layers. As convolutions during forward/backward passes can now operate on low bitwidth weights respectively, DoReFa-Net can accelerate CNNs in both training and inference. A sample algorithm to quantize 32bit weights $\boldsymbol{w}$ to binary weights $\boldsymbol{w^b}$ is also proposed in [33], it is defined as:

$$w^b = E\left(|w|\right) \times sign\left(w\right) \tag{1}$$

To solve the problem of low representation ability of binary networks, ternary weight networks (TWNs) is proposed in [34]. Weights in TWNs are limited to $+1$, 0, and -1. Through minimize the Euclidean distance between the full precision weight $\boldsymbol{W}$ and the ternary weight $\boldsymbol{W^t}$, TWNs have near or exceed the accuracy of 32-bit networks. The ternary precision counterpart has a stronger expression

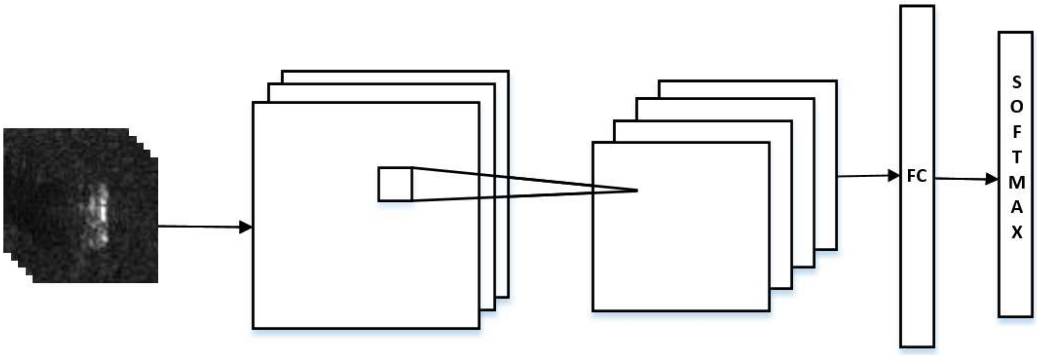

FIGURE 1. The framework of MCNN.

ability than the recently proposed binary precision counterpart, and thus is more efficient than the binary precision counterpart.

Network quantization can greatly compress the network. However, when CNNs are deep, the quantization operation will greatly reduce the classification accuracy of network. Therefore, network quantization is not suitable for use on deep convolutional neural networks (DCNNs).

Knowledge distillation is a useful method to compress CNNs. In [24], Hinton proposed KD and a student-teacher paradigm as a framework of KD. Subsequently, KD was widely used. The work in [35] address model compression by using KD to train thin and deep CNNs, called FitNets, under the supervision of wide and shallower (but still deep) CNNs. This method extends KD so that student networks can be deeper than teacher networks. FitNets mimic the full feature maps of the teacher networks. However, this method does not work when the teacher and the student have significant difference in the network structure. In order to enhance the generalization ability of student networks, the work in [36] adds noise to teacher networks. The above KD extracts knowledge from the bottom layers of the network, while the work in [37] transfer knowledge from the higher hidden layers to students network, as the higher hidden layers include more information. The KD-based algorithm can greatly reduce the parameters needed to construct a network and significantly reduce computational cost.

In order to compress CNNs greatly and retain their accuracy, knowledge distillation can be combined with network quantization. Experiments in [38] prove that it is feasible to combine knowledge distillation and weight quantization. Quantized distillation (QD) in [38] chooses a trained state-of-the-art deep model as the teacher network and designs a small network which has similar structure with teacher as student network. Then, QD quantizes the student network with variable bit-width while training the student network via knowledge distilled. 2bit,4bit and 8bit quantization methods are used to validate this idea on different dataset. Experimental results of QD show that this method has a good performance at 4-bit and 8bit quantization, but loss of accuracy is catastrophic at 2bit.

III. PROPOSED CNN AND ITS TRAINING ALGORITHM

In this section, we first introduce the framework of MCNN for SAR target recognition. Then the training algorithm of MCNN is described.

A. THE FRAMEWORK OF MCNN

The framework of MCNN is shown in Fig. 1. It consists of a convolutional layer and a fully connected layer. The size of convolutional kernel in the convolutional layer is 5×5, and 32 channels is outputted by the convolutional layer. Traditional CNN usually uses convolution kernels of 3×3 to extract optical image features. SAR images have some problems such as speckle and low image clarity. The effect of speckle noise is sometimes very large, which makes it inappropriate to 3×3 convolution kernels in SAR image recognition. A bigger receptive field can reduce the influence of speckle to some extent. Therefore, we uses 5×5 convolution kernels in MCNN to extract features.

The convolutional layer multiplies its input data x by its different convolution kernel weights W, and outputs different feature maps whose number is the same as the number of convolution kernels. The processing of convolution can be formulated as:

$$y_{i',j',k'} = f\left(b'_k + \sum_{i=1}^{h}\sum_{j=1}^{w}\sum_{d=1}^{c} W_{i,j,d,k'} \times x_{i+i',j+j',d}\right) \quad (2)$$

where y is the output, x is input, i', j', k' and d are the coordinate values of the convolution kernel or output. h is the height of convolutional kernel, w is the width of convolutional kernel, c is depth of convolutional kernel. $f(\cdot)$ is activation function. In MCNN, rectified liner unit (Relu) is used as the activation function, and it is formulated as:

$$y_{i,j,k} = max\left(0, x_{i,j,k}\right) \quad (3)$$

After the convolution operation, pooling operation is used to reduce the size of the feature maps, and the size of the pooling windows is 2×2, and it can be formulated as:

$$y_{i',j',k} = \max_{1<i<h,1<j<w} x_{i+i',j+j',k} \quad (4)$$

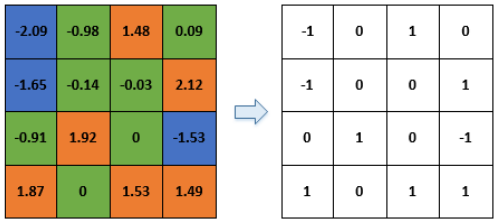

FIGURE 2. A convolutional kernel(right) of 4 × 4 and its ternary form(left).

where h is the height of the pooling window, w is the width of the pooling window.

The fully-connected layer contains 10 nodes. Finally, the processing of softmax is used to get probability distribution of targets, and it can be formulated as:

$$y_{i,j,k} = \frac{exp\left(x_{i,j,k}\right)}{\sum_{d=1}^{c} exp\left(x_{i,j,d}\right)} \tag{5}$$

The parameters of the network can be fine-tuned according to the task requirements.

Method in [34] is used to quantize our student network. This method quantizes all 32bit weights $\mathbf{W}$ in network to ternary weight $\mathbf{W^t}$ in the following ways:

$$W_i^t = f_t\left(W_i \mid \Delta\right) = \begin{cases} 1, & \text{if } W_i > \Delta \\ 0, & \text{if } |W_i| \leqq \Delta \\ -1, & \text{if } W_i < -\Delta \end{cases} \tag{6}$$

Here Δ is a positive threshold parameter. A ternary weight needs 2 bits of memory to store.

The process of quantization aims at minimizing the Euclidian distance between $\mathbf{W}$ and $\mathbf{W^t}$ as follow:

$$\begin{cases} a^*, W^t = \underset{a, W^t}{arg\,min}\ J\left(a^*, W^t\right) = \left\|W - aW^t\right\|_2^2 \\ s.t\ a \geqq 0, \quad W_i^t \in \{-1, 0, 1\},\ i = 0, 1, 2 \cdots \end{cases} \tag{7}$$

where a is the quantization threshold.

For example, Fig. 2 shows a convolutional kernel of 4 × 4 and its ternary form. When ternary network is being trained, both forward pass and backward are ternary weight, and weights in network are updated directly on ternary weight with stochastic gradient descent (SGD). On the hardware platform, the multiplication operation between ternary weights can be transformed into a simple bit computation by special algorithm or hardware processing, which greatly accelerates the inference of network.

B. THE TRAINING ALGORITHM OF MCNN

The proposed training algorithm is based on KD in [24]. Traditional KD algorithms use large trained networks as teacher networks to supervise the training process of student networks. The output vector after the softmax processing of the teacher networks is used as knowledge to guide the learning process of student networks, and this vector is called soft target. The real labels of targets are called hard targets. The training of traditional CNNs only use hard

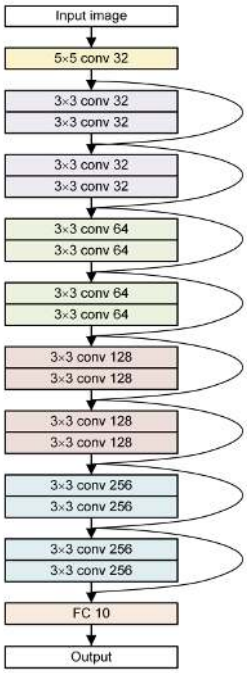

FIGURE 3. The framework of MCNN's teacher network.

targets. Large CNNs have strong learning ability, therefore they can be trained well only under the supervision of hard targets. The learning ability of shallow networks are weak, and they cannot obtain strong ability of recognition only by the supervision of hard targets. As the proposed MCNN only has 2 layers, it's hard to get useful information about the target by itself. Therefore, MCNN is trained by KD-based algorithm and the state-of-the-art network is SAR image recognition is used to as its teacher network. This network is proposed in [27]. The framework of the teacher network in shown as Fig. 3. This teacher network is based on ResNet, and the only difference is the size of kernels in their first convolutional layer.

The traditional knowledge distillation algorithm is improved by us to train the MCNN. The novel algorithm is named gradual distillation (GD). Similar to GD, Quantization distillation (QD) in [38] uses a trained network as the teacher network to train a 32 bit student network, and then quantifies the 32 bit student to 2bit. This indirect way will result in loss of network's accuracy. What's more, 2bit student networks are difficult to learn directly from the hard and soft target vectors. The experimental results in [38] show that this method is not appropriate to train 2bit quantized network.

In GD, teacher network is untrained. Teacher network and student network are trained together at the same time instead of one after another. In each step of training, the teacher network uses its output to guide the students network learning. When student network is trained, a 2bit weight in it changes directly to another 2bit value. In this way, the student network can learn knowledge from the teacher network step by step, and a better route to reduce loss can be found. The flowchart

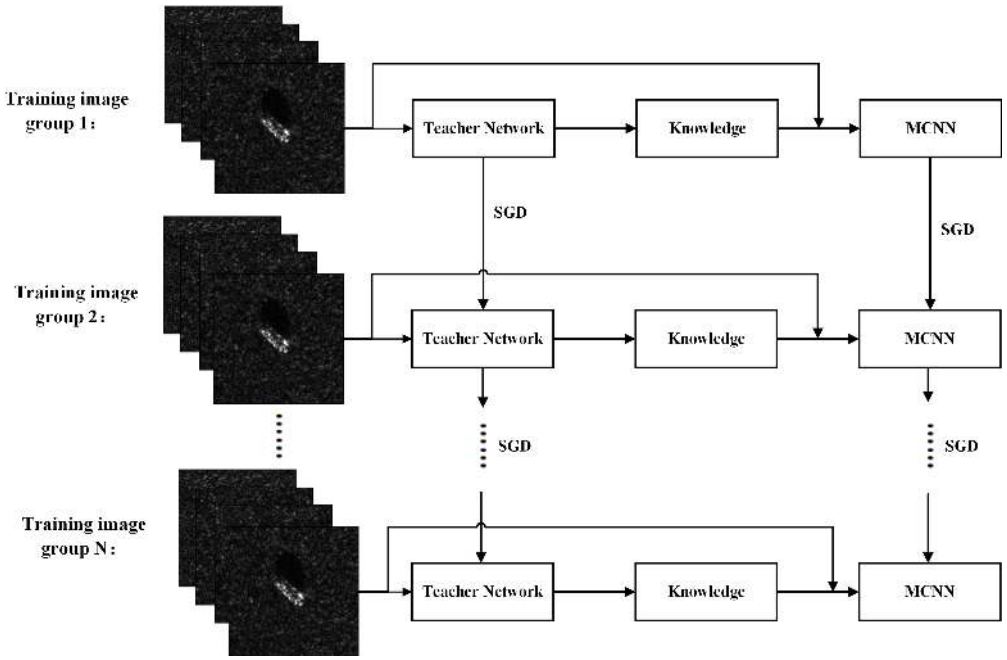

FIGURE 4. The flowchart of GD.

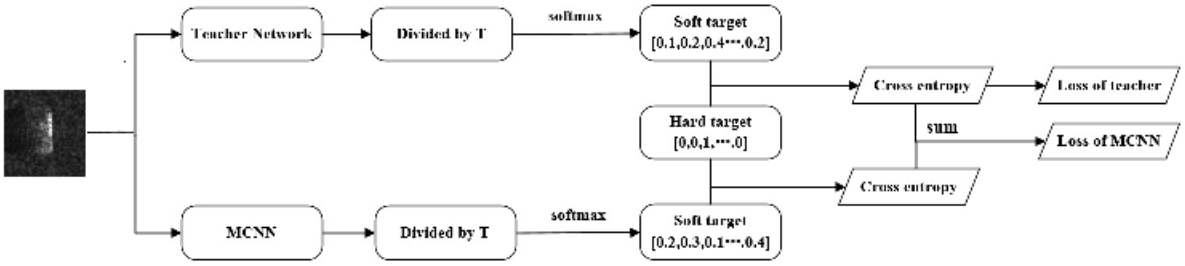

FIGURE 5. The calculation process of the teacher network's loss and student network's loss in GD.

of GD is shown in Fig. 4. The calculation process of the teacher network's loss and student networks's loss in GD is shown in Figure 5. The transferred knowledge is the teacher network's output vector called soft target. True label is the hard target.

The teacher work use the cross entropy between its output and hard target as loss to update its weights, and it can be formulated as:

$$Loss_t = \sum_{i=1}^{n} p\,(i) \times \log \frac{1}{q_t\,(i)} \tag{8}$$

where p is the hard target and $q_t\,(i)$ is the output of the teacher network. n is the length of vector outputted by the teacher network, i.e the type numbers of target.

This loss function of student network contains two parts: one is the cross entropy between student network's output and hard target, and the other is the cross entropy between student network's output and teacher network's target, which can be formulated as:

$$Loss_s = \sum_{i=1}^{n} p\,(i) \times \log \frac{1}{q_s\,(i)} + \sum_{i=1}^{n} q_t\,(i) \times \log \frac{1}{q_s\,(i)} \tag{9}$$

T in Fig. 5 is distilling temperature defined in [24], and it controls the difficulty of the student networks' learning process. The bigger the T, the harder the learning process of the student network.

The training process of GD ends when $Loss_s$ approaches the minimum that it can reach. After be trained by GD, the MCNN is a fully functional network, and it can replace the teacher network in all systems that use the teacher network as a module or work independently.

IV. EXPERIMENT AND ANALYSIS

In this section, experimental dataset and relevant settings are introduced firstly. Then we introduce our experimental results and analyze the outcome results.

TABLE 1. Number of training and test samples from MSTAR dataset.

Category	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL131	ZU234
Trianing	299	233	298	256	233	299	298	232	299	299
Test	274	195	274	195	196	274	173	296	299	299

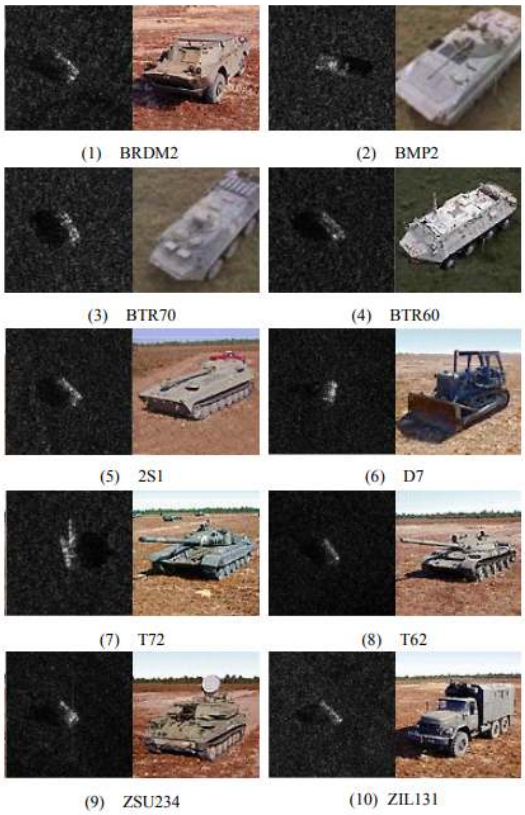

FIGURE 6. Ten types of military targets in SAR images (left) and their corresponding optical images (right).

A. DATASET AND SETTINGS

In this paper, all experiments are conducted on MSTAR dataset. MSTAR is used as a standard data set for testing and comparing SAR classification performance of different models. It is provided by the Air Force Research Laboratory and the Defense Advanced Research Projects Agency (AFRL/DARPA). AFRL/DARPA collected SAR images of hundreds of military targets, but only published a small number of images for research. Ten categories of the public released images is included (2S1, BMP2, BRDM2, BTR60, BTR70, D7, T62, T72, ZIL131, ZU234). In this paper, we follow the official advice and use images with 17° depression as training set and images with 15° depression as test set. The numbers of each class are displayed in Table 1 Ten types of military targets in SAR images (left) and their corresponding optical images (right) are shown in Fig. 6.

As shown in Fig. 6, two objects with large difference in optical image are difficult to distinguish from human eyes in the SAR image. Because of the low imaging resolution, the contour of the target in MSTAR is not obvious. The irrevocable speckle noise and shadows caused by the imaging angle also make it difficult to classify these images.

We conduct all experiments on a desktop with intel i7-8700K processor and a Nvidia Geforce GTX 1080Ti GPU card. The operating system is Linux with cuDNN v9.

As we know, in the absence of data enhancement and multi-branch structure, the CNN in [27] is the art-of-the-state network for SAR image recognition, so it is used as the teacher network. MCNN is the student network. In our experiments, we reproduce the network in [27] and achieve the same result. Student networks which are quantified and trained in traditional methods will be used to compare with MCNN. All student networks has the same framework as MCNN as shown in Fig .1. We use the error rate to measure the performance of a network, which is defined as:

$$Error\ Rate = 1 - OA \tag{10}$$

where OA is the overall accuracy.

B. EXPERIMENTAL RESULTS AND ANALYSIS

1) COMPARISONS BETWEEN MCNN AND TRADITIONAL MACHINE LEARNING

Firstly, the classification obfuscation matrix of MCNN on MSTAR dataset and the accuracy of each class are shown in in Table 2. As shown in Table 2, MCNN has a strong ability to recognize SAR targets, and the error rate is only 1.8%. Confusion is concentrated between similar goals, for example, T62 and T72, BTR60 and BTR70.

In Table 3, We compare MCNN with several traditional machine learning methods, including Decision tree (DT), Logistic regression (LG), Gradient boosting decision tree (GBDT), Multi-Layer perceptron (MLP), Random forest (RF), K-NearestNeighbor (KNN), Support vector machine (SVM) and Naive Bayes (NB).

As shown in Table 3, the error rate of MCNN is much lower than all traditional machine learning methods' error rates. Even the SVM with the lowest error rate in machine learning methods has 1.77 times the error rate of MCNN. Therefore, we can conclude that MCNN is superior to traditional machine learning methods.

2) COMPARISONS BETWEEN MCNN AND SOME EXISTING DCNNS

Firstly, we compare the accuracy of MCNN with some deep neural networks in recent two years. We reproduce some advanced DCNNs for SAR image recognition in several papers and obtain similar experimental results with

TABLE 2. Classification obfuscation matrix of MCNN on MSTAR.

Test set	Classification results										Classification accuracy (%)
	2S1	BMP2	BRDM2	BTR60	BTR70	D7	T62	T72	ZIL131	ZU234	
2S1	269	3	0	0	0	0	1	1	0	0	98.2%
BMP2	5	183	0	1	0	0	3	2	0	1	93.9%
BRDM2	1	0	267	2	3	0	1	0	0	0	97.8%
BTR60	1	0	4	182	7	0	0	1	0	0	93.3%
BTR70	1	0	1	4	188	0	1	0	1	0	96.0%
D7	0	0	0	1	0	272	0	1	0	0	99.3%
T62	0	0	0	0	0	0	173	1	0	0	99.3%
T72	0	0	0	0	0	0	2	294	0	0	99.3%
ZIL131	0	0	0	0	1	0	0	0	297	1	99.3%
ZU234	0	0	0	0	1	0	0	1	1	296	98.9%
Average classification accuracy: 98.2%											

TABLE 3. Comparisons between MCNN and traditional machine learning.

Methods	Error rate (%)
MCNN	1.80
SVM	3.18
RF	3.51
KNN	4.66
GBDT	4.83
MLP	6.64
LG	9.86
NB	17.7
DT	29.3

TABLE 4. Comparisons between MCNN and some existing CNNs.

Network	Error rate (%)
ResNet-18 in [27]	1.2
MCNN	1.8
Network in [4]	1.7
Network in [39]	1.92
Network in [40]	3.3
Network in [41]	4.05

TABLE 5. Comparisons between MCNN and the network in [30].

Network	Error rate (%)	parameter	Model size
MCNN	1.8	62k	248KB
Network in [31]	8.0	1.035M	412KB

original papers. The results on the MSTAR dataset of comparison is shown in Table 4.

As shown in Table 4, in addition to ResNet-18(the teacher network of MCNN) and network in [4], MCNN has the lowest error rate, which proves that MCNN is superior to most DCNNs.

Then, we compare MCNN with the lightweight network proposed by [30], and results are shown in Table 8.

As shown in Table 5, compared to the lightweight network in [30], MCNN's error rate is 6.2% smaller. What's more, MCNN has smaller model size and less parameter to process. These two points prove that MCNN is better than the network in [30].

TABLE 6. Error rate (%) of teacher, KD and MCNN.

Category	Teacher	MCNN	KD
2S1	3.4	1.8	3.3
BMP2	3.1	6.1	3.1
BRDM2	0	2.2	4.0
BTR60	5.7	6.7	8.2
BTR70	0.5	4.0	3.6
D7	0.7	0.7	6.7
T62	1.1	0.7	2.6
T72	2.1	0.7	1.6
ZU131	0.4	0.7	0.4
ZU234	0	1.1	1.1
TOTAL	1.2	1.8	2.7

3) COMPARISONS BETWEEN MCNN'S TRAINING ALGORITHM AND TRADITIONAL KD

In order to prove that the training algorithm of MCNN, i.e proposed QD, is better than traditional KD, comparative experiments are conducted, and the results are shown as Table 6. The accuracy of each target category and total accuracy on MSTAR are shown. KD is the abbreviation of the ternary student network trained by traditional knowledge distillation algorithm. Teacher is the abbreviation of the teacher network. The hyperparameter T in Fig. 5 is set to 2.

As shown in Table 6, the total error rate of MCNN is 1.8%, almost the same as teacher network, only 0.6% more than the accuracy of teacher network. In some categories, such as T62 and T72, the error rate of MCNN is even less than that of teacher network. Compared with KD, the total error rate has been reduced by 1/3. The result shows that the QD has a better performance than KD.

What's more, MCNN converges faster than the ternary network trained by traditional distillation through the curves of accuracy during training as shown in Fig. 7. The faster convergence speed of MCNN and higher accuracy prove that our knowledge distillation algorithm is superior to the traditional knowledge Distillation Algorithm in training shallow networks. In Fig. 7, the size of a training batch is 128. It means in each round of training, we feed 128 images to the network.

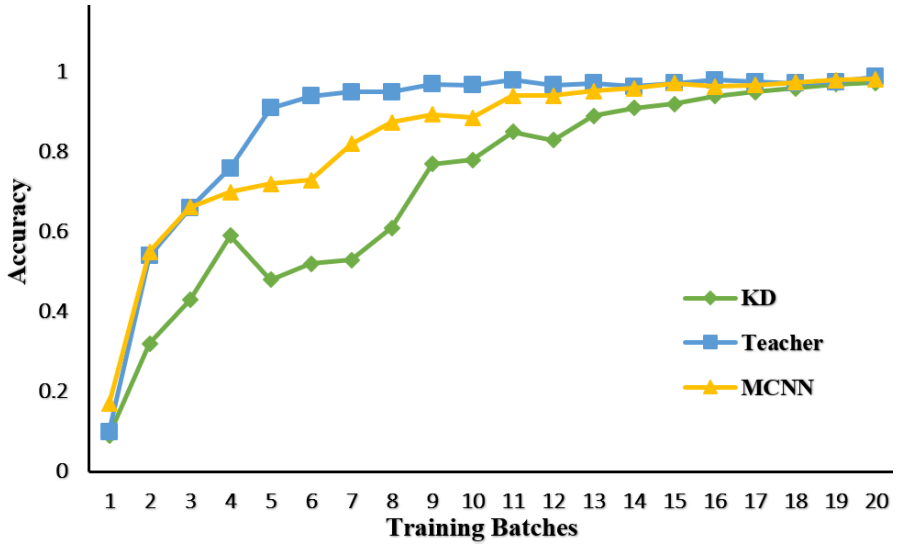

FIGURE 7. The Accuracy Trend of Teacher, MCNN and KD.

TABLE 7. Comparisons between students networks.

Networks	Error rate (%)
MCNN	1.8
KD	2.7
QD	4.0
TS	4.2
FS	4.8

4) COMPARISONS BETWEEN STUDENTS NETWORKS

To illustrate our compression effect and the role of GD, the error rates of the networks which training independently and the student network trained by knowledge distillation are shown in Table 7. In Table 7, QD is the abbreviation of 2-bit student network trained by QD in [38]. TS is the abbreviation of ternary student network without knowledge distillation, and FS is the abbreviation of full-precision student network without knowledge distillation.

As shown in Table 7, MCNN has the lowest error rate. Compared with TS and MCNN, QD reduces the error rate of TS by 2.3 times. Compared to 2-bit student network trained by QD, the error rate of MCNN is 2.2 times less. All of these prove that QD can effectively improve the recognition ability and generalization ability of shallow networks, and MCNN is 2-bit network, with the same number of parameters, it needs 16 times less storage than 32-bit networks, and the speed of calculation between 2-bit weights is also much faster than 32-bit weights on the embedded devices.

5) COMPARISONS BETWEEN MCNN AND ITS TEACHER NETWORK

Teacher network and student network training are synchronized, and the training time is 172.2s. The two networks' inference time for the test set which including 2425 images is shown in Table 8. The inference time of MCNN is 7.2 times shorter than that of teachers' network. The error rate, parameters, model size, and multiplications to recognize an image of the teacher network and MCNN are also shown in Table 8. The 32-bit teacher network has about 11 million parameters, and it takes up 44 MB memory. The MCNN network has 62 thousand parameters which only needs 248 KB memory. The model size of MCNN is about 177 times smaller than the teacher network. What's more, 103.9 million multiplications are needed in teacher network to sort an SAR image, while MCNN only needs 8.1 million multiplications, which is 12.8 times less.

TABLE 8. Comparisons between MCNN and its teacher network.

Network	Error rate (%)	Params	Model size	Multiplications	Inference time
MCNN	1.8	62k	248KB	8.1M	0.7s
Teacher	1.2	11M	44MB	103.9M	5.06s

6) RESULTS UNDER EXTENDED OPERATING CONDITION

It is well known that SAR image recognition is greatly affected by depression angle. In order to prove that our network has strong generalization ability, experiments are conducted under the condition of extended operating condition (EOC) of MSTAR dataset. EOC includes four categories: T72, 2S1, BRDM2, ZSU-234. According to the official recommendation, we use the images with 17° degrees as training set, and the SAR images with 30° as test set. The information of training set and test set is given in Table 9. The experimental results are shown in Table 10.

As shown in Table 10, although the training set and the test set are quite different in depression angle, MCNN still maintains a low error rate which is similar as its teacher network. This shows that MCNN has strong generalization ability.

TABLE 9. Number of training and test samples of EOC.

Category	Training set(17°)	Test set(30°)
T72	232	288
2S1	299	288
BRDM2	298	287
ZSU-234	299	288
Total	1128	1151

TABLE 10. The results of the experiment under EOC.

Category	Teacher	MCNN
T72	7.6	8.7
2S1	5.2	5.9
BRDM	0.7	1.0
ZSU-234	1.4	1.7
Total	3.7	4.3

V. CONCLUSIONS

Because of the over-parameterization and high computational cost, traditional deep convolutional neural networks are difficult to deploy in real-time recognition system of SAR sensors. Therefore, a micro CNN and gradual distillation are proposed in this paper. Compressed by ternary quantization, the proposed MCNN takes up much less memory (248kB) and requires only 8.1 million multiplications to figure out the target class in SAR images. Following a teacher-student paradigm, gradual distillation makes MCNN faster with more accurate SAR target recognition than most machine learning methods and existing CNNs. The experimental results show that the MCNN can achieve 98.2% on the MSTAR dataset, which verify the effective performance of the proposed method.

From the above, MCNN is small, fast and accurate, which makes it possible to deploy it in real-time recognition systems of SAR sensors. Meanwhile, the research results of this paper can promote the application of complex deep network structure in the borne radar recognition system.

REFERENCES

[1] T. Zhang, J. Liang, Y. Yang, G. Cui, L. Kong, and X. Yang, "Antenna deployment method for multistatic radar under the situation of multiple regions for interference," *Signal Process.*, vol. 143, pp. 292–297, Feb. 2018.

[2] J. Wu, W. Pu, Y. Huang, J. Yang, and H. Yang, "Bistatic forward-looking SAR focusing using ω–K based on spectrum modeling and optimization," *IEEE J. Sel. Topics Appl. Earth Observat. Remote Sens.*, vol. 11, no. 11, pp. 4500–4512, Nov. 2018.

[3] L. Peng, X. Liu, M. Liu, L. Dong, M. Hui, and Y. Zhao, "SAR target recognition and posture estimation using spatial pyramid pooling within CNN," in *Proc. Int. Conf. Opt. Instrum. Technol., Optoelectron. Imag./Spectrosc. Signal Process. Technol.*, vol. 10620, Jan. 2018, Art. no. 106200W.

[4] Z. Tian, L. Wang, R. Zhan, J. Hu, and J. Zhang, "Classification via weighted kernel CNN: Application to SAR target recognition," *Int. J. Remote Sens.*, vol. 39, no. 23, pp. 9249–9268, 2018.

[5] Y. Kwak, W.-J. Song, and S.-E. Kim, "Speckle-noise-invariant convolutional neural network for SAR target recognition," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 4, pp. 549–553, Apr. 2019.

[6] P. Liu, K.-K. R. Choo, L. Wang, and F. Huang, "SVM or deep learning? A comparative study on remote sensing image classification," *Soft Comput.*, vol. 21, no. 23, pp. 7053–7065, 2017.

[7] A. El Housseini, A. Toumi, and A. Khenchaf, "Deep Learning for target recognition from SAR images," in *Proc. Seminar Detection Syst. Archit. Technol. (DAT)*, Feb. 2017, pp. 1–5.

[8] F. Zhang, C. Hu, Q. Yin, W. Li, H.-C. Li, and W. Hong, "Multi-aspect-aware bidirectional LSTM networks for synthetic aperture radar target recognition," *IEEE Access*, vol. 5, pp. 26880–26891, 2017.

[9] Z. Huang, Z. Pan, and B. Lei, "Transfer learning with deep convolutional neural network for SAR target classification with limited labeled data," *Remote Sens.*, vol. 9, no. 9, p. 907, 2017.

[10] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. (2016). "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size." [Online]. Available: https://arxiv.org/abs/1602.07360

[11] A. G. Howard *et al.* (2017). "MobileNets: Efficient convolutional neural networks for mobile vision applications." [Online]. Available: https://arxiv.org/abs/1704.04861

[12] Y. Cheng, D. Wang, P. Zhou, and T. Zhang. (2017). "A survey of model compression and acceleration for deep neural networks." [Online]. Available: https://arxiv.org/abs/1710.09282

[13] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Proc. Adv. Neural Inf. Process. Syst.*, 1990, pp. 598–605.

[14] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Proc. Adv. Neural Inf. Process. Syst.*, 1993, pp. 164–171.

[15] Y. Guo, A. Yao, and Y. Chen, "Dynamic network surgery for efficient DNNS," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1379–1387.

[16] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2754–2761.

[17] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1269–1277.

[18] V. Vanhoucke, A. Senior, and M. Z. Mao, "Improving the speed of neural networks on CPUs," in *Proc. Deep Learn. Unsupervised Feature Learn. NIPS Workshop*, vol. 1, Dec. 2011, p. 4.

[19] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proc. Int. Conf. Mach. Learn.*, Jun. 2015, pp. 1737–1746.

[20] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio. (2016). "Binarized neural networks: Training deep neural networks with weights and activations constrained to $+1$ or -1." [Online]. Available: https://arxiv.org/abs/1602.02830

[21] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Oct. 2016, pp. 525–542.

[22] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2654–2662.

[23] G. Hinton, O. Vinyals, and J. Dean, "Dark knowledge," presented at the Keynote BayLearn, vol. 2, 2014.

[24] G. Hinton, O. Vinyals, and J. Dean. (2015). "Distilling the knowledge in a neural network." [Online]. Available: https://arxiv.org/abs/1503.02531

[25] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7130–7138.

[26] S. Zagoruyko and N. Komodakis. (2016). "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer." [Online]. Available: https://arxiv.org/abs/1612.03928

[27] H. Furukawa. (2017). "Deep learning for target classification from SAR imagery: Data augmentation and translation invariance." [Online]. Available: https://arxiv.org/abs/1708.07920

[28] J. Ding, B. Chen, H. Liu, and M. Huang, "Convolutional neural network with data augmentation for SAR target recognition," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 3, pp. 364–368, Mar. 2016.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. the IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[30] H. Chen, F. Zhang, B. Tang, Q. Yin, and X. Sun, "Slim and efficient neural network design for resource-constrained SAR target recognition," *Remote Sens.*, vol. 10, no. 10, p. 1618, 2018.

[31] S. Han, H. Mao, and W. J. Dally. (2015). "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding." [Online]. Available: https://arxiv.org/abs/1510.00149

[32] X. Lin, C. Zhao, and W. Pan, "Towards accurate binary convolutional neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 345–353.
[33] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou. (2016). "DoReFa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients." [Online]. Available: https://arxiv.org/abs/1606.06160
[34] F. Li, B. Zhang, and B. Liu. (2016). "Ternary weight networks." [Online]. Available: https://arxiv.org/abs/1605.04711
[35] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. (2014). "FitNets: Hints for thin deep nets." [Online]. Available: https://arxiv.org/abs/1412.6550
[36] B. B. Sau and V. N. Balasubramanian. (2016). "Deep model compression: Distilling knowledge from noisy teachers." [Online]. Available: https://arxiv.org/abs/1610.09650
[37] P. Luo, Z. Zhu, Z. Liu, X. Wang, and X. Tang, "Face model compression by distilling knowledge from neurons," in *Proc. AAAI*, Feb. 2016, pp. 3560–3566.
[38] A. Polino, R. Pascanu, and D. Alistarh. (2018). "Model compression via distillation and quantization." [Online]. Available: https://arxiv.org/abs/1802.05668
[39] Y. Xue, J. Pei, Y. Huang, J. Yang, and Y. Zhang, "Target recognition for SAR images based on heterogeneous CNN ensemble," in *Proc. IEEE Radar Conf. (RadarConf)*, Apr. 2018, pp. 0507–0512.
[40] F. Gao, Z. Yue, J. Wang, J. Sun, E. Yang, and H. Zhou, "A novel active semisupervised convolutional neural network algorithm for SAR image recognition," *Comput. Intell. Neurosci.*, vol. 2017, Aug. 2017, Art. no. 3105053.
[41] L. Peng, M. Liu, X. Liu, L. Dong, M. Hui, and Y. Zhao, "SAR image classification based on CNN in real and simulation datasets," in *Proc. 9th Int. Conf. Graphic Image Process. (ICGIP)*, vol. 10615, 2018, Art. no. 106152V.

RUI MIN (M'11) received the B.E., M.S., and Ph.D. degrees in circuits and systems from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2000, 2003, and 2012, respectively.

From 2014 to 2015, he was a Visiting Scholar with the Department of Electrical Engineering, University of Texas at Arlington. He is currently an Associate Professor with the School of Information and Communication Engineering, UESTC. His research interests include radar signal processing, image processing, and compressive sensing. He is a Reviewer of the IEEE TRANSACTIONS ON FUZZY SYSTEMS and the IEEE SENSORS JOURNAL.

HAI LAN (S'18) received the B.E. degree in electronic science and technology from the University of Electronic Science and Technology of China, Chengdu, China, in 2017, where he is currently pursuing the M.S. degree in signal processing. His research interests include machine learning, and image processing.

ZONGJIE CAO (M'10) received the B.E. and Ph.D. degrees from Xi'an Jiaotong University, Xi'an, China, in 1999 and 2005, respectively.

From 2006 to 2008, he was a Postdoctoral Researcher with the Communication and Information System Postdoctoral Center, University of Electronic Science and Technology of China (UESTC). In 2008, he joined the School of Electronic Engineering, UESTC, where he is currently a Professor with the School of Information and Communication Engineering. He has authored or coauthored over 50 papers. His current research interests include radar signal processing, synthetic aperture radar imaging, and target recognition. He is a Reviewer of several international journals and conferences, such as the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING, the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, the *IET Radar, Sonar & Navigation*, and *Remote Sensing*.

ZONGYONG CUI (S'13–M'15) received the B.E. and Ph.D. degrees in signal and information processing from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2007 and 2015, respectively. From 2013 to 2014, he was a Visiting Student with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. He is currently a Lecturer with the School of Information and Communication Engineering, UESTC. His research interests include radar image processing, target recognition, and machine learning. He is a Reviewer of the IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, the *IET Radar, Sonar & Navigation*, and the *Journal of Electronic Imaging*.