Semi-Supervised Deep Transfer Learning-Based on Adversarial Feature Learning for Label Limited SAR Target Recognition

WEI ZHANG, YONGFENG ZHU, AND QIANG FU
National Key Laboratory of Science and Technology on ATR, Institution of Electronic Science, National University of Defense Technology, Changsha 410073, China
Corresponding author: Yongfeng Zhu (zhuyongfeng@nudt.edu.cn)

ABSTRACT The data-driven convolutional neural networks (CNNs) have achieved great progress in Synthetic Aperture Radar automatic target recognition (SAR-ATR) after being trained in a large scale of labeled samples. However, the insufficiency of labeled SAR data always leads to over-fitting, causing significant performance degradation. To solve the mentioned problem, a semi-supervised transfer learning method based on generative adversarial networks (GANs) is presented in the present paper. The discriminator of GAN with an encoder and a discriminative layer is redesigned to make it capable of learning the feature representation of input data with unsupervised settings. Instead of training a deep neural network with the insufficient labeled data set, we first train a GAN with varieties of unlabeled samples to learn generic features of SAR images. Subsequently, the learned parameters are readopted to initialize the target network to transfer the generic knowledge to specific SAR target recognition task. Lastly, the target network is fine-tuned using both the labeled and unlabeled training samples by a semi-supervised loss function. We evaluate the proposed method on the MSTAR and OpenSARShip data set with 80%, 60%, 40%, and 20% of the training set labeled, respectively. The results suggest that the proposed method achieves up to 23.58% accuracy enhancement over the random-initialized model.

INDEX TERMS SAR-ATR, convolutional neural networks (CNNs), semi-supervised transfer learning, adversarial feature learning.

I. INTRODUCTION
Deep convolutional neural networks (CNNs) have achieved progress in synthetic aperture radar (SAR) target recognition [1], for its robust ability to learn high-level features. However, this type of method requires large labeled data set to train the model, whereas most of the SAR data sets are unlabeled or sparsely labeled, leading to severe overfitting when a deep CNN is being trained. As manually labeling SAR data sets is hard to achieve or time-consuming, how to improve the performance of CNN trained in label-limited training set has become a hot topic in this area.

Chen et al. [2] proposed an all-convolutional network (A-ConvNet) replacing all full-connected layers with the convolution layers. This method reduces overfitting by reducing model parameters and achieves better performance than that of the general CNN in the classification of the Moving and Stationary Target Acquisition and Recognition (MSTAR) benchmark dataset [3]. However, it remains a data-hungry method. Its performance decreases significantly with reducing labeled training samples.

Transfer learning emerged as a popular training method to overcome the label-limited difficulty by initializing the model using the parameters that have been learned from a large data set [4]–[6], and then fine-tuning the initialized model with small amounts of labeled samples of the target task. [7] fine-tuned a CNN pre-trained in CIFAR-10 optical data set for SAR images classification and achieved better performance than the random-initialized CNN. Reference [8] pre-trained an unsupervised generative network, Auto-Encoder (AE), to learn feature representations of large amounts of unlabeled SAR scene images. Subsequently, a classification layer was added at the end of the encoder and the whole network was fine-tuned. This method enhanced the performance of the CNN trained in small labeled training set. Besides, the model pre-training requires no additional label.

The associate editor coordinating the review of this manuscript and approving it for publication was Nizam Uddin Ahamed.
However, the above methods only exploit the labeled training samples in target domain, while the information of unlabeled samples in target domain is wasted.

In the meantime, deep semi-supervised learning is a powerful framework leveraging unlabeled training samples to assist in training deep neural networks under insufficient labeled samples [9]–[11]. The common implementation of semi-supervised learning is to exploit the labeled samples to train a supervised CNN to predict the labels of the unlabeled samples. Then, the predicted labels are adopted to train the CNN with the unlabeled samples. Some other methods also employ GANs to achieve semi-supervised learning [12]. It has been applied to enhance the performance of SAR target recognition with label limitation [13]. The essential of semi-supervised learning is to use unlabeled samples to estimate the distribution of the data. One critical assumption in many semi-supervised learning algorithms is the structure assumption: samples within the same structure (e.g., a cluster or a manifold) are likely to have the same label [14]. However, SAR images change significantly with the azimuth and depression angles. SAR images in the identical class can be quite different, which makes that some unlabeled training samples may not satisfy the structure assumption.

Given the mentioned discussions, a semi-supervised transfer learning method is proposed to overcome the labeled limited difficulty of SAR images. As numerous unlabeled SAR images are available, the unsupervised generative network is first utilized to learn feature representations from the data based on the idea of [8]. Nevertheless, we propose an improvement that learns features using a variant of more advanced generative adversarial networks (GANs) [15], instead of the Auto-Encoder, as shown in Figure 1. Afterwards, the learned parameters are leveraged to initialize a CNN and an adaptive layer is added as a feature classifier. Lastly, the initialized model is fine-tuned using both the labeled and unlabeled samples of our target task through a semi-supervised loss. The proposed method achieves great progress in the classification of MSTAR and OpenSARShip data set when the labeled samples are insufficient. Even only 20% of training samples are labeled, the method can still achieve an accuracy over 90% on the MSTAR data set.

The rest of this letter is organized as follow. Section II introduces the implementation of our method. Section III demonstrates the experiment settings. Section IV discusses the performance of the experimental results. Section V gives the conclusion.

II. METHOD IMPLEMENTATION

According to the concept of transfer learning, the training data is split into a source domain and a target domain. In the present paper, the source domain is composed of varieties of unlabeled SAR data $X_s$ that is different form but associated with the data to be classified. The target domain is the specific SAR data set that contains labeled samples $\{X_t, Y_t\}$ and unlabeled samples $\tilde{X}_t$ waiting for classification. As illustrated in Figure 1, the overall architecture is divided into the source network and the target network. The source network is first pre-trained with both $X_s$ and $\tilde{X}_t$ to learn feature representations. Subsequently, the target network is initialized using the pre-trained parameters of the source network to transfer the learned features to the classification of the target domain. And lastly, fine-tune the initialized model using $\tilde{X}_t$ and $\{X_t, Y_t\}$ through a semi-supervised loss function.

A. ADVERSARIAL FEATURE LEARNING

To learn features from the unlabeled samples, an unsupervised feature learning method is adopted based on the generative adversarial network (GAN).

The typical GAN is generally known as a powerful framework to generate high-quality data for its adversarial training method. It learns a generative network $G$ to “fool” the discriminative network $D$ by mapping an arbitrary latent feature distribution $C$ to the data as close to the real data as possible. Meanwhile, the discriminative network $D$ is trained to distinguish between the generated data and the real data as significantly as possible. $G$ and $D$ check and balance each other through alternately optimizing the following two adversarial objective functions [15]:

$$
\min_D L_{AdvD} = -E_{c \sim C} \log [1 - D(G(c))] - E_{x \sim \{X_s, \tilde{X}_t\}} \log [D(x)]. \quad (1)
$$

$$
\min_G L_{AdvG} = -E_{c \sim C} \log [D(G(c))]. \quad (2)
$$

where $D$ produces a normalized scalar, indicating the probability that the data is real. Numerous experiments have
demonstrated that the adversarial training method helps GANs generate more realistic data and be more generalized than the Auto-Encoder (AE). Nevertheless, the original GAN framework cannot map the data to the latent feature space, since it lacks a component like the encoder in AE. References [18] and [19] solve the problem by adding an additional encoder component to the original GAN, which requires more memory to restore the whole network. Thus, we directly do some modifications to the general GAN architecture, decomposing it into an encoder $E$ and a discriminative layer $T$, as shown in the source network of Figure 1. $E$ maps the generated data $G(c)$ and the real data $x$ to the feature vectors $c' \sim C'$, where $c'$ has the same dimension as the latent feature vectors $c \sim C$. And $T$ discriminates whether $c'$ is extracted from the real data or the generated data. Since we expect to recover data from $c'$, $c'$ should be highly correlated with $c$. Thus, we add a reconstruction loss term $L_{rec}$ that measures the correlation between $c$ and $c'$ to the original adversarial objective functions, as shown below:

$$
\min_{T,E} L_{AdvD} = -\mathbb{E}_{c \sim C} \log [1 - T(E(G(c)))]
- \mathbb{E}_{x \sim \{X, \tilde{X}\}} \log[T(E(x))] + \lambda_1 L_{rec},
$$

$$
\min_{E} L_{AdvE} = -\mathbb{E}_{c \sim C} \log[T(E(G(c)))] + \lambda_2 L_{rec},
$$

where $\lambda_1$ and $\lambda_2$ are hyper-parameters that control the bias of each term.

In mathematics, the inner product is often used to reflect the angle between two vectors. If and only if the angle between the two vectors is zero, their inner product reaches a maximum. At this point, the two vectors are linearly dependent. Thus, we naturally use the inner product of the normalized vectors of $c$ and $c'$ to represent the correlation between them. Consider the input feature vectors of the form $c = [c_1, c_2, \ldots, c_n]$ and the predicted feature vectors of the form $c' = [c'_1, c'_2, \ldots, c'_n]$, the reconstruction loss can be defined as

$$
L_{rec} = -\mathbb{E}_{c \sim C, c' \sim C'} \{ \sum_{i=1}^{n} \frac{c_i - \text{avg}(c)}{\|c\|_2} \cdot \frac{c'_i - \text{avg}(c')}{\|c'\|_2} \},
$$

where $\mu = [\text{avg}(c), \ldots, \text{avg}(c)]$, $\mu' = [\text{avg}(c'), \ldots, \text{avg}(c')]$, $\text{avg}(\cdot)$ is the statistical average, and $\|\cdot\|_2$ is the $L_2$-norm. By this means, we can learn an encoder that maps the generated data $G(c)$ to the similar feature space to $C$ by minimizing $L_{rec}$. As the well-trained $G$ can successfully confuse $D, X_s, \tilde{X}$ and $G(c)$ are embedded into the same distribution of the feature space by $E$. Further, we can assume that $E$ extracts common features from $X_s$ and $\tilde{X}$, which is of great significance to alleviate overfitting.

B. SEMI-SUPERVISED KNOWLEDGE TRANSFER

The result of current experiments suggests that the pre-trained models can still retain the generic features of the data even after being fine-tuned many iterations [17]. Thus, we compose the target model with the pre-trained encoder $E$ and a random-initialized adaptive layer $A$ to transfer the generic knowledge to the specific classification task, as shown in Figure 1. $A$ outputs an $n$-dimensional vector normalized by the SoftMax function to represent the probability of each category, where $n$ denotes the category number of the target task. Subsequently, we fine-tune the whole model using both $\{X_t, Y_t\}$ and $\tilde{X}_t$ to adapt it to the target task through a semi-supervised loss function:

$$
\min_{A,E} L_{semi} = L_{sup} + \lambda(t)L_{unsup},
$$

where $L_{sup}$ is the supervised loss, $L_{unsup}$ is the unsupervised loss, $\lambda(t)$ is a coefficient balancing the two terms. $L_{sup}$ can be simply defined as the CrossEntropy:

$$
L_{sup} = -\mathbb{E}_{x_t \sim X_t, y \sim Y_t} y \cdot \log[A(E(x_t))],
$$

where $y$ is a one-hot encoding vector that records the real label of $x_t$.

According to the “cluster assumption” [20], samples of the same cluster have a high probability of belonging to the same class. Thus, we can consider the predicted category as the soft-label of the unlabeled samples. Meanwhile, the “manifold assumption” [14] suggests that the unlabeled samples should help to make the data space of the same class denser so that the decision function can better fit the data. Based on the above analysis, we use information entropy as the unsupervised loss to regularize the unlabeled samples, shown as:

$$
L_{unsup} = -\mathbb{E}_{x_t \sim \tilde{X}_t} A(E(\tilde{X}_t)) \cdot \log[A(E(\tilde{X}_t))],
$$

where $A(\cdot)$ and $E(\cdot)$ denote the forward propagation functions of the adaptive layer and the encoder, respectively. The coefficient $\lambda(t)$ should be a tiny value at the beginning to reduce the disturbance of the unlabeled sample to the supervised training. Then, $\lambda(t)$ increases gradually with iterations to use benefit from unlabeled samples. Thus, the form of $\lambda(t)$ is given as follow:

$$
\lambda(t) = \frac{\lambda_3}{1 + e^{-\frac{t}{T}}},
$$

where $\alpha$ and $T$ refer to hyper-parameters controlling the horizontal scale and the translation of the function, respectively. And $\lambda_3$ is important for the network performance. If it is overly large, it will disturb the training of labeled data. If $\lambda_3$ is too small, the semi-supervised learning cannot fully leverage the unlabeled data.

III. EXPERIMENT

This section interprets the experiment settings employed for verifying the performance of our method.

A. NETWORK CONFIGURATIONS FOR EXPERIMENT

A DCGAN-based [21] architecture is adopted to build both encoder $E$ and generator $G$ of the GAN for experiments. Thus, the hidden layers of both $E$ and $G$ primarily consist of convolution kernels, of which the forward propagation is
TABLE 1. Each component architecture of our model.

| Encoder (E)                   | Generator (G)                |
|-------------------------------|------------------------------|
| **Layer Configuration**       | **Layer Configuration**      |
| Input size 128 × 128 × 1      | Input code 128               |
| Conv1 4 × 4 @64               | Dense Node × 16384          |
| BatchNormalization           | TransConv1 4 × 4 @1024      |
| LeakyReLU                     | BatchNormalization          |
| Conv2 4 × 4 @128              | ReLU                         |
| BatchNormalization           | TransConv2 4 × 4 @512       |
| LeakyReLU                     | BatchNormalization          |
| Conv3 4 × 4 @256              | ReLU                         |
| BatchNormalization           | TransConv3 4 × 4 @256       |
| LeakyReLU                     | BatchNormalization          |
| Conv4 4 × 4 @512              | ReLU                         |
| BatchNormalization           | TransConv4 4 × 4 @128       |
| LeakyReLU                     | BatchNormalization          |
| Conv5 4 × 4 @1024             | ReLU                         |
| BatchNormalization           | TransConv5 4 × 4 @64        |
| LeakyReLU                     | BatchNormalization          |
| Dense Node × 128              | ReLU                         |
| Discriminative layer (T)     | TransConv6 4 × 4 @251      |
| Dense Node × 1                | tanh                        |
| Sigmoid                       |                               |
| Adaptive layer (A)            |                               |
| Dense Node × n_classes        |                               |
| Softmax                       |                               |

expressed as:

\[ O_j(w, h) = \sum_{i=1}^{N} \sum_{u,v=0}^{K-1} W^{(j)}_{i}(u, v) F^{(i)}_{j}(w-u, h-v), \]  \hspace{1cm} (10)

where \( O_j(w, h) \) denotes the output of the \( j \)th kernel, \( F^{(i)}_{j}(w-u, h-v) \) refer to the pixel at the position \( (w-u, h-v) \) of \( i \)th feature map of the \( l \)th layer. And \( W^{(j)}_{i}(u, v) \) denotes the trainable convolution kernel with the size of \( K \times K \). For simplicity, the size of all the convolution and transpose convolution kernels is set to \( 4 \times 4 \), while the stride in each dimension is set to two pixels.

The discriminative layer \( T \) is a dense-connected layer producing a scalar activated by a sigmoid function. The target network consists of the encoder \( E \) and the adaptive layer \( A \) whose outputs are normalized by the SoftMax function. The input size of both source domain and target domain is set to \( 128 \times 128 \). Thus, the model architecture is provided as presented in Table 1, where \( n_{classes} \) denotes the number of categories that the target SAR data set contains. The convolution kernel is expressed as “height × width @channels”. The input codes \( c \) are set to 128-dimensional noise vectors of which the elements are produced randomly by a Gaussian distribution \( c \sim N(0,1) \). The entire network covers 4,257,792 trainable parameters in total.

As GANs are relatively difficult to train, we exploit some empirical techniques to accelerate and stabilize the training. First, LeakyReLU acts as the activation function of \( E \) to avoid the activation value falling in the interval where the gradient is zero, which is expressed as:

\[ \text{LeakyReLU}(x) = \max(0.2x, x). \]  \hspace{1cm} (11)

For comparison, the general ReLU function is shown as follows:

\[ \text{ReLU}(x) = \max(0, x). \]  \hspace{1cm} (12)

Moreover, we add a BatchNormalization layer [22] between the convolution kernels and the activation functions to stabilize the training of GANs. And then, the adaptive gradient-based Adam optimizer [23] is utilized for training the model.

B. SOURCE DOMAIN DATA SET

In this paper, we’ve obtained large-scale unlabeled SAR images collected by the TerraSAR-X, Sentinel-1 and Gaofen-3 remote sensing satellites. The obtained SAR images are adopted to compose the source domain data set of the transfer learning. To fit the input size, the original image is cropped into \( 128 \times 128 \) pixels with overlap. Slices containing several strong scattering points are mainly selected to form the training set of the source domain, as they are more similar to the SAR target images. The total source domain contains more than 20,000 image slices involving multiple scenes and resolutions. Some samples of the source training set are presented in Figure 2.

FIGURE 2. Samples of the source domain training set.

C. TEST IN THE MSTAR DATA SET

One of the accuracy tests for the target task is performed using the airborne Moving and Stationary Target Acquisition and Recognition(MSTAR) system. To comprehensively assess the performance, the proposed algorithm is tested both under standard operating conditions (SOC) and EOC. In the SOC setup, the experimental data set contains ten types of objects, which are 2S1, ZSU234, BMP2, BRDM2, BTR60, BTR70, D7, ZIL131, T62, and T72. Some samples of MSTAR data set and their corresponding optical images are presented in Figure 3. The same class of targets in the training and testing sets has the same serial number but differs in azimuth and depression angle. Images of training set are obtained at \( 17^\circ \) depression angle, and images of test set are acquired at \( 15^\circ \) depression angle. Each sample in the MSTAR data set consists of \( 128 \times 128 \) pixels.
SAR images are always easily affected by the variance of depression angles. In the EOC setup, the proposed method is assessed with respect to large depression angle variations. The open source MSTAR data set provide four classes of targets(2S1, BRDM-2, T-72, and ZSU-234) that contain samples obtained at 30° depression angle. Thus, the proposed method is tested on these samples, with the training set chosen from the corresponding four targets in the training set of SOC.

Some existing methods use numerous data augmentation techniques to enlarge the training set and crop the test set into small patch to achieve high performance. In the experiments of this paper, the raw images of MSTAR without any preprocessing and data enhancement are fed directly into the target neural network to present the robustness of the proposed method. The proportions of labeled samples to the training set are set to 80%, 60%, 40%, and 20%, respectively.

**D. TEST IN THE OPENSARSHIP DATA SET**

We also assess the proposed method on OpenSARShip data set [24]. This data set provides several types of ship chips collected from the Sentinel-1A/B satellite. Compared with MSTAR, this data set is obtained directly from the natural scene. The environment is more complex, which brings more challenges to the recognition algorithm. In the evaluation, four major classes (cargo ship, fishing ship, tanker and tug) of ships are chosen for experiment. For each class, 500 SAR ship chips are collected. As the categories of fishing ship or tug only contain about 120 samples, chips of these two categories are translated and rotated to create some novel data. Then, the data set was randomly split into training set and test set according to a ratio of 3:2. As the original chips are of different sizes, we resized them to 128 × 128 pixels using bilinear interpolation. Figure 4 shows some samples of the four classes of ships in the OpenSARShip data set.

**E. OTHER DETAILS**

To elucidate the improvement of our algorithm, four control trails, i.e., supervised learning (SL), semi-supervised learning (SSL), supervised transfer learning (STL), and our semi-supervised transfer learning (SSTL) are set up. The specific implementations are shown as follows:

1) Supervised learning (SL): directly training the random-initialized target network using only labeled samples through the supervised loss \( L_{sup} \).

2) Semi-supervised learning (SSL): directly training a random-initialized target network using both labeled and unlabeled samples through the semi-supervised loss \( L_{semi} \).

3) Supervised transfer learning (STL): training the target network which is initialized by the pre-trained source network using only labeled samples through the supervised loss \( L_{sup} \).

4) Semi-supervised transfer learning (SSTL): training the target network which is initialized by the pre-trained source network using both labeled and unlabeled samples through the semi-supervised loss \( L_{semi} \).

Furthermore, we also compared the performance of our method with the state-of-art semi-supervised learning and transfer learning methods for SAR target recognition, including transfer learning based on Auto-Encoder(TL-AE) [8], semi-supervised learning based on GAN (SSGAN) [13]. As these methods are implemented using different network architectures, and test on the images cropped into different sizes in their own experiments. To make it comparable, we do not use the original network, but evaluate all the methods with the same target network architecture as TABLE 1 presents. In the mean time, the input images are all regulated to 128 × 128-pixel.

We set the learning rate of both \( L_{AdvD} \) and \( L_{AdvG} \) to 1e-4 for adversarial feature learning, and 1e-5 of \( L_{semi} \) for...
generic knowledge transfer. The mini-batch of each iteration is 64 for the limitation of memory. \( \lambda_1 \) and \( \lambda_2 \) can be adjusted during training the GAN. These two parameters should first ensure the convergence of \( L_{\text{rec}} \), and at the same time should be as small as possible to avoid slowing down GAN training. In this paper, they are set as: \( \lambda_1 = \lambda_2 = 2 \). The hyper-parameters \( T \) and \( \alpha \) primarily aim to provide a smooth transition for supervised learning and semi-supervised learning to reduce the fluctuation during training. We adjust these two hyper-parameters according the change of \( L_{\text{semi}} \) and finally set \( T = 100 \), \( \alpha = 5 \). \( \lambda_3 \) is the critical hyper-parameters for the network performance and is related to the labeled rate. This parameter is determined using cross-validation for different labeled rates, as shown in TABLE 2.

**TABLE 2.** \( \lambda_3 \) for different labeled rates.

| Labeled rate | 80% | 60% | 40% | 20% |
|--------------|-----|-----|-----|-----|
| \( \lambda_3 \) | 0.4 | 0.4 | 0.3 | 0.2 |

## IV. RESULTS

The experimental results are presented in two parts. In the first part, the reliability of the features learned from the source domain is to be proved by data reconstruction. In the second part, we will check the improvement of the proposed method in the MSTAR and OpenSARShip data set, respectively.

### A. RECONSTRUCT IMAGES WITH LEARNED FEATURES

A common method to verify the feasibility of features extracted from data is to see whether the data can be reconstructed with fidelity by these features. We enter a SAR image Figure 5a into the encoder to produce a feature vector and feed the feature vector into the generator to reconstruct the image. The reconstructed image is given in Figure 5c. It is suggested the reconstructed image is almost identical to the real image. Besides, the encoder is utilized to produce a feature vector of another image Figure 5b. Subsequently, images can be generated using the linear interpolations between these two vectors, as shown in Figure 5d-5j. The generated images represent the transition between Figure 5a and Figure 5c, which demonstrates the linear continuity of the extracted features.

### B. ACCURACY ON MSTAR

In this subsection, the classification accuracies of semi-supervised transfer learning (SSTL) and the other three comparison models are given. To ensure that all of the methods are adequately trained, we keep training the models until their accuracies of the labeled training samples stabilize at 100% and the loss values reach dynamic equilibrium. Though the training accuracy has been stable, the test accuracy still fluctuates during the training due to the different depression angle between the training and test sets [25]. To mitigate the impact of this fluctuation, the result is recorded as pairs of mean and standard deviation (\( \mu \pm std \)) over the last twenty epochs.

**TABLE 3.** The recognition accuracy on test set for SOC (80%-labeled training set).

| Category  | SL (%) | STL(%) | SSL (%) | SSTL(ours) (%) |
|-----------|--------|--------|---------|---------------|
| 2S1       | 97.08±0.26 | 97.96±0.00 | 93.27±0.18 | 97.21±0.28 |
| BMP2      | 99.64±0.13 | 98.85±0.00 | 98.07±0.16 | 99.57±0.00 |
| BRDM      | 98.97±0.00 | 100±0.00 | 99.49±0.00 | 100±0.00 |
| BTR70     | 68.42±2.56 | 94.21±0.36 | 77.43±0.88 | 93.13±0.03 |
| BTR80     | 96.84±0.00 | 99.27±0.31 | 98.90±0.00 | 98.91±0.00 |
| D7        | 87.41±0.70 | 87.13±0.38 | 88.26±0.60 | 97.58±0.23 |
| T62       | 92.31±0.00 | 94.87±0.00 | 90.12±0.22 | 98.33±0.00 |
| T72       | 81.73±1.61 | 93.58±1.14 | 85.33±0.88 | 97.98±0.54 |
| ZIL131    | 98.47±0.11 | 99.05±0.00 | 96.96±0.44 | 98.39±0.18 |
| ZSU234    | 100±0.00 | 100±0.00 | 100±0.00 | 100±0.00 |
| Average   | 92.21±0.39 | 96.12±0.19 | 93.03±0.25 | 98.39±0.23 |

**TABLE 4.** The recognition accuracy on test set for SOC (60%-labeled training set).

| Category  | SL (%) | STL(%) | SSL (%) | SSTL(ours) (%) |
|-----------|--------|--------|---------|---------------|
| 2S1       | 95.89±3.45 | 95.51±0.20 | 95.27±0.6 | 92.64±2.22 |
| BMP2      | 96.46±0.35 | 98.50±0.11 | 98.68±0.5 | 98.81±0.32 |
| BRDM      | 98.41±0.47 | 100±0.05 | 98.78±0.1 | 100±0.00 |
| BTR70     | 49±4.80 | 81.59±0.37 | 79.98±1.49 | 93.74±0.59 |
| BTR80     | 97.65±0.75 | 98.17±0.00 | 98.52±0.27 | 99.05±0.18 |
| D7        | 77.74±2.86 | 87.02±0.23 | 85.54±1.12 | 94.74±1.34 |
| T62       | 85.31±6.36 | 92.31±0.00 | 89.90±0.34 | 97.36±0.39 |
| T72       | 62.75±1.61 | 76.28±0.98 | 80.40±1.53 | 94.47±1.35 |
| ZIL131    | 95.92±1.30 | 98.26±0.25 | 98.55±0.00 | 98.58±0.62 |
| ZSU234    | 100±0.00 | 100±0.00 | 100±0.00 | 100±0.00 |
| Average   | 85.22±1.79 | 92.89±0.18 | 92.39±0.65 | 96.49±0.74 |

**TABLE 5.** The recognition accuracy on test set for SOC (40%-labeled training set).

| Category  | SL (%) | STL(%) | SSL (%) | SSTL(ours) (%) |
|-----------|--------|--------|---------|---------------|
| 2S1       | 54.81±3.81 | 69.14±0.37 | 67.87±0.18 | 75.83±0.83 |
| BMP2      | 95.47±0.21 | 96.35±0.00 | 97.84±0.22 | 98.76±0.17 |
| BRDM      | 94.01±1.2 | 98.47±0.00 | 94.41±1.11 | 99.46±0.11 |
| BTR70     | 40.59±8.0 | 66.76±1.06 | 73.00±1.49 | 84.30±2.23 |
| BTR80     | 96.37±4.05 | 98.18±0.00 | 96.25±0.37 | 99.27±0.00 |
| D7        | 77.74±0.3 | 82.31±0.72 | 85.46±0.52 | 93.71±0.87 |
| T62       | 74.44±1.52 | 85.56±1.18 | 87.28±0.00 | 95.76±0.00 |
| T72       | 64.8±4.13 | 66.85±1.40 | 76.95±0.14 | 89.84±0.96 |
| ZIL131    | 83.46±1.30 | 92.78±0.18 | 89.51±0.29 | 97.40±0.10 |
| ZSU234    | 99.96±0.11 | 100±0.00 | 100±0.00 | 100±0.00 |
| Average   | 79.8±0.39 | 85.83±0.34 | 87.47±0.43 | 93.07±0.34 |

**FIGURE 5.** Reconstruct SAR images with GAN, where (b) is the reconstructed image of (a). (d)-(j) present the reconstructed images of linear interpolations between (a) and (c).
methods in the four experiments, and the superiority of our method becomes more noticeable as the labeled training set decreases. Even when only 20% of training samples are labeled, our algorithm still achieves an accuracy of nearly 90%. The performance enhancement exceeds more than the addition of that of the semi-supervised learning (SSL) and supervised transfer learning (STL).

The test accuracy for EOC setup is listed in TABLE 7-10. Figure 6 presents the average test accuracy of each experiment for EOC setting. It is still found that the accuracy of SSTL is to some extent enhanced compared with other methods. But in this part of experiments, STL always achieves better performance than that of SSL. This indicates that when the training set and the test set are quite different, the performance enhancement of transfer learning is more significant than that of semi-supervised learning.

C. ACCURACY ON OPENSARSHIP

The test accuracy on the OpenSARShip data set is listed in TABLE 11-14. And Figure 8 visualizes the results of the four methods. Given the results, it is therefore concluded that SSTL always achieves the optimal accuracy. For the study of this paper, the transfer learning (SSTL) method significantly outperforms the semi-supervised learning method (SSL),
TABLE 13. The recognition accuracy on test set of OpenSARShip (40%-labeled training set).

| Category | SSL (%) | STL (%) | SSL (%) | SSTL(out) (%) |
|----------|---------|---------|---------|---------------|
| Cargo    | 70.10±0.29 | 66.71±0.29 | 71.22±0.37 | 88.89±1.37 |
| Fishing  | 71.42±0.05 | 76.5±0.31 | 72.68±0.73 | 84.64±0.54 |
| Tanker   | 64.97±1.17 | 90.12±0.21 | 69.70±0.43 | 91.57±0.54 |
| Tug      | 77.52±2.35 | 86.97±0.11 | 76.08±0.81 | 88.07±1.08 |
| Average  | 71.01±0.29 | 85.08±0.09 | 72.42±0.14 | 88.29±0.35 |

TABLE 14. The recognition accuracy on test set of OpenSARShip (20%-labeled training set).

| Category | SSL (%) | STL (%) | SSL (%) | SSTL(out) (%) |
|----------|---------|---------|---------|---------------|
| Cargo    | 68.05±0.13 | 76.09±0.25 | 66.18±0.39 | 83.52±0.39 |
| Fishing  | 66.92±0.18 | 71.45±0.87 | 69.48±0.11 | 80.05±0.92 |
| Tanker   | 49.60±0.43 | 84.74±0.33 | 61.58±0.75 | 83.59±0.29 |
| Tug      | 71.80±0.29 | 87.81±0.63 | 74.78±0.54 | 85.07±0.64 |
| Average  | 64.09±0.11 | 80.02±0.17 | 68.01±0.19 | 83.06±0.20 |

D. COMPARISON WITH STATE-OF-ART METHODS.

TABLE 15. The recognition accuracy of SSL, SSGAN, STL, TL-AE, and our semi-supervised transfer learning method (SSTL).

| Labeled rate | SSL | SSGAN | STL | TL-AE | SSTL(out) |
|--------------|-----|-------|-----|-------|-----------|
| 80%          | 93.03±0.25 | 92.39±0.65 | 87.47±0.43 | 78.87±0.36 |
| 60%          | 93.02±0.46 | 92.41±0.29 | 87.86±0.39 | 78.58±0.59 |
| 40%          | 96.12±0.19 | 92.59±0.63 | 85.83±0.34 | 76.34±0.54 |
| 20%          | 95.99±0.27 | 92.68±0.29 | 85.11±0.44 | 75.99±0.49 |

REFERENCES

[1] Y. Li, J. Wang, Y. Xu, H. Li, Z. Miao, and Y. Zhang, “DeepSAR-Net: Deep convolutional neural networks for SAR target recognition,” in Proc. IEEE 2nd Int. Conf. Big Data Anal. (ICBDA), Beijing, China, Mar. 2017, pp. 740–743.

[2] S. Chen, H. Wang, F. Xu, and Y. Q. Jin, “Target classification using the deep convolutional networks for SAR images,” IEEE Trans. Geosci. Remote Sens., vol. 54, no. 8, pp. 4800–4817, Aug. 2016.

[3] E. R. Keydel, S. W. Lee, and J. T. Moore, “MSTAR extended operating conditions: A tutorial,” Proc. SPIE, vol. 2757, pp. 228–242, Jun. 1996.

[4] D. Malmgren-Hansen, A. Kusk, J. Dall, A. A. Nielsen, R. Engholm, and H. Skriver, “Improving SAR automatic target recognition models with transfer learning from simulated data,” IEEE Geosci. Remote Sens. Lett., vol. 14, no. 9, pp. 1484–1488, Sep. 2017.

[5] H. Lang, S. Wu, and Y. Xu, “Ship classification in SAR images improved by AIS knowledge transfer,” IEEE Geosci. Remote Sens. Lett., vol. 15, no. 3, pp. 439–443, May 2018.

[6] Y. Wang, C. Wang, and H. Zhang, “Combining a single shot multibox detector with transfer learning for ship detection using sentinel-1 SAR images,” in Proc. SAR Big Data Era, Models, Methods Appl. (BIGSAR-DATA), Beijing, China, Nov. 2017, pp. 1–4.

[7] C. Kang and C. He, “SAR image classification based on the multi-layer network and transfer learning of mid-level representations,” in Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS), Beijing, China, Jul. 2016, pp. 1146–1149.

[8] Z. Huang, Z. Pan, and B. Lei, “Transfer learning with deep convolutional neural network for SAR target classification with limited labeled data,” in Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS), Atlanta, Georgia, vol. 3, Jun. 2013, p. 2.
[13] F. Gao, Y. Yang, J. Wang, J. Sun, E. Yang, and H. Zhou, “A deep convolutional generative adversarial networks (DCGANs)-based semi-supervised method for object recognition in synthetic aperture radar (SAR) images,” in *Remote Sens.*, vol. 10, no. 6, p. 846, May 2018.

[14] J. Weston, F. Ratle, H. Mobahi, and R. Collobert, “Deep learning via semi-supervised embedding,” in *Neural Networks: Tricks Trade*. Berlin, Germany: Springer, 2012, pp. 639–655.

[15] I. Goodfellow, J. Pouget-Abadie, and M. Mirza, “Generative adversarial networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, 2014, pp. 2672–2680.

[16] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, “Analysis of representations for domain adaptation,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Vancouver, BC, Canada, 2006, pp. 137–144.

[17] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Montreal, QC, Canada, 2014, pp. 3320–3328.

[18] J. Donahue, P. Krähenbühl, and T. Darrell, “Adversarial feature learning,” May 2016, arXiv:1605.09782. [Online]. Available: https://arxiv.org/abs/1605.09782

[19] X. Chen, Y. Duan, R. Houthooft, J. Schulman, L. Sutskever, and P. Abbeel, “InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Barcelona, Spain, 2016, pp. 2172–2180.

[20] O. Chapelle and A. Zien, “Semi-supervised classification by low density separation,” in *Proc. Int. Workshop Artif. Intell. Statist. (AISTATS)*, Barbados, Caribbean, vol. 2005, pp. 57–64, Jan. 2005.

[21] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” Nov. 2015, arXiv:1511.06434. [Online]. Available: https://arxiv.org/abs/1511.06434

[22] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating deep network training by reducing internal covariate shift,” Feb. 2015, arXiv:1502.03167. [Online]. Available: https://arxiv.org/abs/1502.03167

[23] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” Dec. 2014, arXiv:1412.6980. [Online]. Available: https://arxiv.org/abs/1412.6980

[24] L. Huang, B. Liu, B. Li, W. Guo, W. Yu, Z. Zhang, and W. Yu, “OpenSARShip: A dataset dedicated to Sentinel-1 ship interpretation,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 1, pp. 195–208, Jan. 2018.

[25] D. A. E. Morgan, “Deep convolutional neural networks for ATR from SAR imagery,” *Algorithms Synth. Aperture Radar Imagery*, vol. 9475, p. 94750F, May 2015.

**WEI ZHANG** was born in Qingdao, China, in 1995. He received the B.E. degree in electronics from Beihang University, Beijing, China, in 2017. He is currently pursuing the M.S. degree in information and communication engineering with the National University of Defense Technology (NUDT), Changsha, China. His research interests include applications of transfer learning and deep learning algorithms to automatic radar target recognition.

**YONGFENG ZHU** received the B.E. degree in electronic engineering from Zhejiang University (ZJU), Hangzhou, China, in 2001, and the Ph.D. degree in information and communication engineering from the National University of Defense Technology (NUDT), in 2009. He is currently an Associate Research Fellow of the ATR Laboratory, NUDT. His research interests include automatic target recognition technology, radar signal, and data processing.

**QIANG FU** was born in Changsha, China, in 1962. He received the B.E. degree in electronics engineering and the Ph.D. degree in information and telecommunication systems from the National University of Defense Technology (NUDT), Changsha, in 1983 and 2004, respectively. He is currently a Professor with the ATR Laboratory, NUDT. His research interest includes radar system design, precise guidance, and target recognition.