Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 7724264, 11 pages
https://doi.org/10.1155/2023/7724264

Research Article

Remote Sensing Image Classification with Few Labeled Data Using Semisupervised Learning

Yibing Li, Sitong Zhang, Xiang Li, and Fang Ye

Key Laboratory of Advanced Marine Communication and Information Technology, Ministry of Industry and Information Technology, College of Information and Communication Engineering, Harbin Engineering University, Harbin, China

Correspondence should be addressed to Fang Ye; yefang0923@126.com

Received 10 May 2022; Revised 25 September 2022; Accepted 6 October 2022; Published 20 April 2023

Academic Editor: Wei Li

Synthetic aperture radar (SAR) as an imaging radar is capable of high-resolution remote sensing, independent of flight altitude, and independent of weather. Traditional SAR ship image classification tends to extract features manually. It relies too much on expert experience and is sensitive to the scale of SAR images. Recently, with the development of deep learning, deep neural networks such as convolutional neural networks are widely used to complete feature extraction and classification tasks, which improves algorithm accuracy and normalization capabilities to a large extent. However, deep learning requires a large number of labeled samples, and the vast bulk of SAR images are unlabeled. Therefore, the classification accuracy of deep neural networks is limited. To tackle the problem, we propose a semisupervised learning-based SAR image classification method considering that only few labeled images are available. The proposed method can train the classification model using both labeled and unlabeled samples. Moreover, we improve the unsupervised data augmentation (UDA) strategy by designing a symmetric function for unsupervised loss calculation. Experiments are carried out on the OpenSARShip dataset, and results show that the proposed method reaches a much higher accuracy than the original UDA.

1. Introduction

The Internet of Things (IoT) supports a smarter society by enabling substantial numbers of sensors to perceive the physical world [1, 2]. Being a form of radar, synthetic aperture radar (SAR) is capable of high-resolution remote sensing. Moreover, due to its all-time, all-weather, and large-scale observation capabilities, the SAR system can provide detailed information about the monitored region and, thus, plays an important role in many applications of both military and civilian fields [3–17]. For example, the SAR can gather systematically high-quality data of a city and help build a smart city [18]. SAR images provide prodigious amounts of information, so efficient classification methods are needed to sort SAR images into different categories.

Traditional methods proposed in [19–22] have a similar structure which requires a lot of manual experience. The feature extraction and classification processes are not flexible and need a high manual cost. With the development of computational hardware and neural networks, deep learning (DL) shows great potential in solving image classification problems. In recent years, convolutional neural networks (CNNs) have been commonly used to automatically extract information from SAR images. In [23], the author proposed a CNN architecture that consists of three convolutional layers and one fully connected layer. Li et al. [24] used a CNN structure to extract features from SAR images and then further cluster these sample features in the feature space using the metric learning model. Zhang et al. [25] proposed an improved CNN model to solve the limited sample issue via feature augmentation and ensemble learning strategies. Chen et al. [26] presented a new all-convolutional network (A-ConvNet), which only consists of sparsely connected layers without fully connected layers being used.

The performance of deep neural networks is highly related to the quality of samples and their labels. However, the majority of SAR images are unlabeled since annotation of SAR data is time-consuming [27]. Besides, unlabeled SAR images still have considerable information available for the training process. Therefore, how to utilize unlabeled

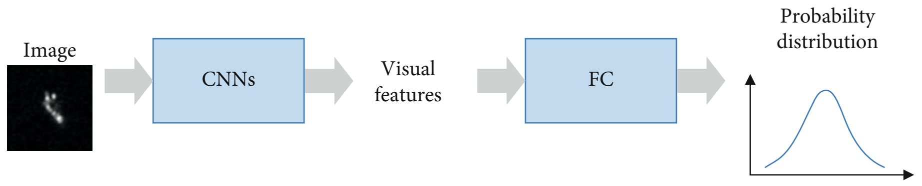


FIGURE 1: Image classification.

samples effectively is an urgent problem to be solved in SAR image classification. In this paper, for the situation that only a few labeled data are available, we propose a semisupervised learning-based method (named UDA-ALE) for SAR image classification. The method utilizes both labeled data and unlabeled data for model training, which can avoid the model overfitting problem due to the lack of labeled samples. And UDA-ALE makes the most of information in unlabeled data, which helps the network model learn faster. In addition, we designed a symmetric function for unsupervised loss calculation, which improves the robustness of the method.

The main contributions of this work are summarized as follows:

(1) We proposed a novel SAR image classification method based on semisupervised learning

(2) We developed a symmetric function for unsupervised loss calculation

(3) We validated the efficiency of the proposed method on the OpenSARShip dataset

The rest of this paper is organized as follows. In Section 2, we introduce the image classification problem, data augmentation, and asymmetric consistency learning. The proposed method is elaborated in Section 3. In Section 4, we describe the experiment settings and analyze the results. Last, we conclude the paper in Section 5.

2. Preliminaries

This section begins with formulating image classification problems and then provides a brief introduction to data augmentation technology and asymmetric consistency learning.

2.1. Image Classification Formulation. Image classification is one of the major topics in the field of artificial intelligence and the IoT. The goal of image classification is to distinguish different types of images according to their characteristics in the image information. Recent works [28–31] have focused on using convolutional neural networks (CNN) to classify images. Many types of CNN models, such as LeNet, VGG, and GoogleNet, are proposed to improve the accuracy of image classification. In general, as illustrated in Figure 1, the CNNs used in image classification usually work in three steps. First, the input image is normalized and resized. Second, single or multiple blocks of convolution layers are used to extract visual features of the image. Third, fully connected layers (FC) map these features to the probability distribution of different categories.

2.2. Data Augmentation. In image classification problems, the training of neural networks entirely depends on data and labels. However, the vast bulk of SAR images are unlabeled, and the labeling process requires a large number of human resources. Therefore, data augmentation [32] is proposed to enlarge the size of datasets, which can be divided into supervised data augmentation and unsupervised data augmentation.

2.2.1. Supervised Data Augmentation. Supervised data augmentation is aimed at obtaining new training samples by performing a series of transformation operations on the original data without changing their labels. Data augmentation, one of transformation operations, can be denoted as $q(\hat{x}|x)$. It obtains the augmented sample $\hat{x}$ based on the original sample x and the data distribution $\hat{x} \sim q(\hat{x}|x)$. Although a large body of supervised data augmentation methods [33–36] improve the performance to some extent, it can be seen that the improvement is limited because augmentation operations are only applied on labeled datasets that are relatively small scale. Therefore, unsupervised data augmentation (UDA) [37] is proposed to enlarge unlabeled datasets. UDA utilizes relatively large-scale unlabeled datasets, which can strive for further improvement of the training performance.

2.2.2. Unsupervised Data Augmentation. UDA is a semisupervised technique in which models are trained on both labeled and unlabeled data. Therefore, it can be adopted to obtain more unlabeled training data in the semisupervised learning framework, and using unlabeled data can make the output smoother. Figure 2 shows the UDA algorithm. First, the backbone network M receives the sample x_1 and outputs the probability distribution $p_\theta(y|x_1)$. θ is the parameter set of M. Then, the label $f^*(x_1)$ and the probability distribution $p_\theta(y|x_1)$ are used to calculate the supervised loss value. On the other side, the unlabeled sample x_2 is input into the $\widehat{M}$, and the probability distribution $p_{\widehat{\theta}}(y|x_2)$ is obtained. $\widehat{\theta}$ is the parameter set of the network $\widehat{M}$. It is obtained by copying from θ in real time and does not update through the back-propagation process. Next, x_2 is added with a little disturbance ε to get $\hat{x} = q(x, \varepsilon)$, which is called the augmentation strategy. After getting the probability distribution $p_\theta(y|\hat{x})$, the UDA calculates the unsupervised loss value. Last, the final loss can be obtained by adding

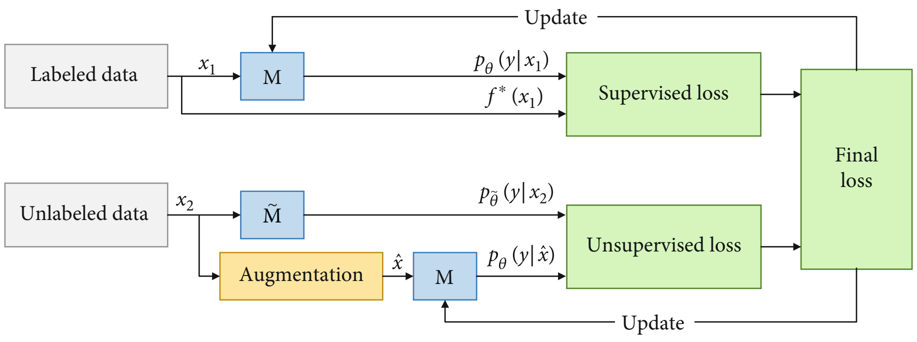


Figure 2: The UDA algorithm.

the supervised loss value and the unsupervised loss value, denoted as

$$\min_\theta J(\theta) = E_{x_1 \sim p_L(x)}[-\log p_\theta(f^*(x_1)|x_1)] + \lambda E_{x_2 \sim p_U(x)} E_{\hat{x} \sim q(\hat{x}|x_2)}\left[D\left(p_{\tilde{\theta}}(y|x_2) \| p_\theta(y|\hat{x})\right)\right], \quad (1)$$

where $p_L(x)$ and $p_U(x)$ represent the data distributions of the labeled samples and the unlabeled samples, respectively. $q(\hat{x}|x_2)$ is the data augmentation operation, and λ is the weight coefficient. $D(\cdot)$ represents the difference between two probability distributions. The parameters of M can be updated by minimizing the final loss. The UDA algorithm makes the backbone network insensitive to disturbance and noise added into input and hidden layers; thus, the output of the backbone network can be smoother. Furthermore, the information of labeled samples can be gradually passed to unlabeled samples by means of final loss minimization. Taken together, UDA improves the performance of the image classification by increasing the diversity of unlabeled data.

2.3. Asymmetric Consistency Learning. According to Figure 2 and Equation (1), the final loss value is calculated by adding the supervised loss and unsupervised loss. Both of them are asymmetric consistent in the UDA algorithm, and the following part describes in greater detail the two loss functions.

2.3.1. Supervised Loss. The supervised loss function can be split from the final loss, shown as

$$E_{x_1 \sim p_L(x)}[-\log p_\theta(f^*(x_1)|x_1)]. \quad (2)$$

In order to make a concrete analysis of the function, we take an iteration as an example. In the iteration, a minibatch of labeled samples is randomly selected, which is denoted as $\{(x_i, y_i)\}_{i=1}^{n_S}$. n_S represents the number of labeled samples in the minibatch. Moreover, the cross-entropy function is used to calculate the loss function, which is asymmetric consistent. Therefore, the difference between the probability distribution and the label can be obtained:

$$E_{x_1 \sim p_L(x)}[-\log p_\theta(f^*(x_1)|x_1)] = -\frac{1}{n_S}\sum_{i=1}^{n_S}\sum_{c=1}^{n_c} \mathbf{1}(y_i == c) \log p_\theta(y_i == c|x_i), \quad (3)$$

where n_c represents the number of categories and c is the category of the current sample. y_i represents the actual label of the current sample. $\mathbf{1}(y_i == c)$ is a flag function. It equals 1 when y_i is c and vice versa.

2.3.2. Unsupervised Loss. Similarly, the unsupervised loss function can be obtained by splitting the final loss, shown as

$$E_{x_2 \sim p_U(x)} E_{\hat{x} \sim q(\hat{x}|x_2)}\left[D\left(p_{\tilde{\theta}}(y|x_2) \| p_\theta(y|\hat{x})\right)\right]. \quad (4)$$

Unlike the supervised loss, the unsupervised loss only includes two probability distributions instead of actual labels. Thus, Kullback-Leibler divergence (KL-divergence) [38], which can measure the distance between two different distributions, is adopted to measure the difference $D(\cdot)$, and it is asymmetric consistent as well. Given two distributions P, Q defined on the probability space $\mathcal{X}$, the KL-divergence is

$$D(P\|Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right). \quad (5)$$

The loss value calculated by KL-divergence gets larger as the difference between the two probability distributions gets greater, and it ranges from 0 to ∞. KL-divergence equals 0 only if the two probability distributions are identical. Therefore, the unsupervised loss can be rewritten as

$$E_{x_2 \sim p_U(x)} E_{\hat{x} \sim q(\hat{x}|x_2)}\left[D\left(p_{\tilde{\theta}}(y|x_2) \| p_\theta(y|\hat{x})\right)\right] = \frac{1}{n_U}\sum_{i=1}^{n_U}\sum_{c=1}^{n_c} p_{\tilde{\theta}}(y_i = c|x_i) \log \frac{p_{\tilde{\theta}}(y_i = c|x_i)}{p_\theta(y_i = c|\hat{x}_i)}, \quad (6)$$

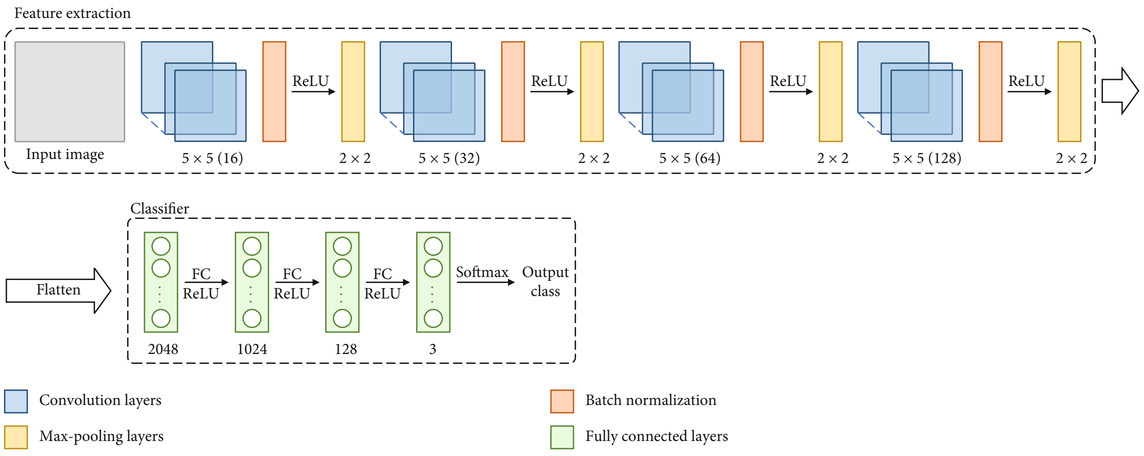


FIGURE 3: The structure of the backbone network for image classification.

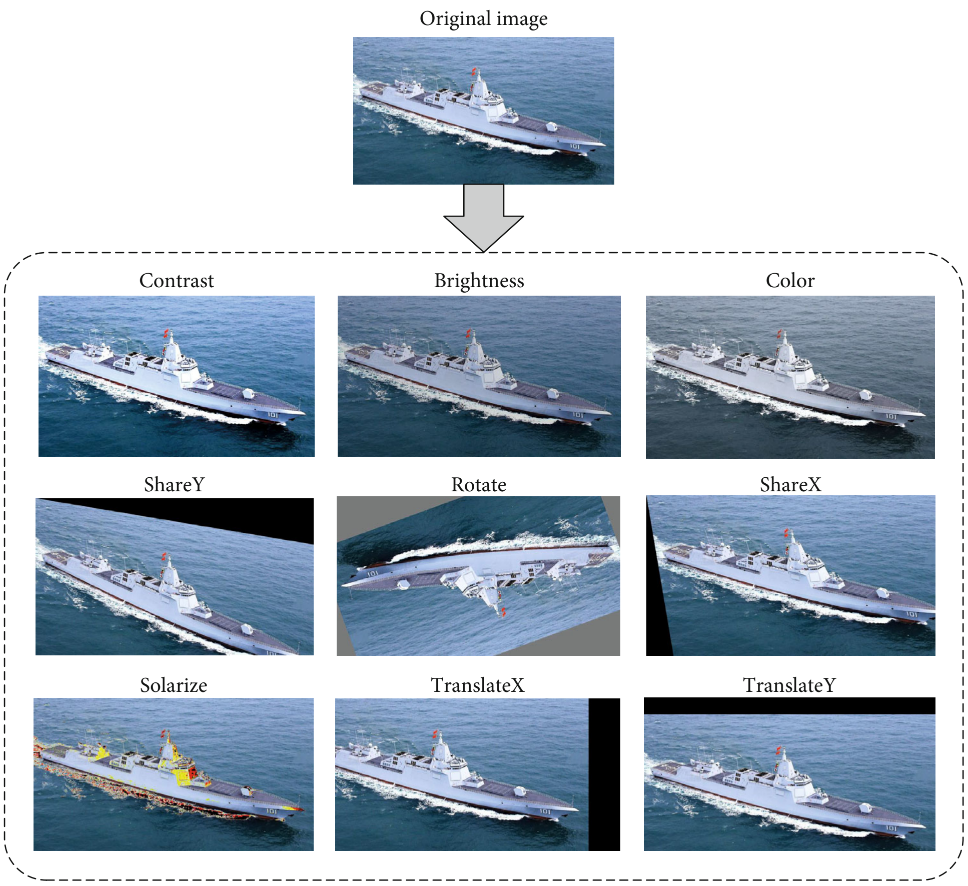


FIGURE 4: Data augmentation operations.

where n_U is the number of unlabeled samples in the mini-batch. $p_{\tilde{\theta}}(y_i = c|x_i)$ is set to the probability distribution of the actual event, and $p_\theta(y_i = c|\hat{x}_i)$ is regarded as the probability distribution of the theoretically fitted event.

3. Proposed Method

In this section, we give a detailed account of the proposed method, including the network structure, data augmentation strategy, symmetric consistency learning, and some additional training techniques.

3.1. Network Structure. The structure of the backbone network is shown in Figure 3. It works in two steps, (1) feature extraction and (2) classification. First, four convolution modules are designed to extract visual features of images, containing network layers, batch normalization layers, rectified linear units (ReLUs), and max-pooling layers. Second,

```
Input: collection of data augmentation operations: (identity, autocontrast, equalize, rotate, solarize, color, posterize, contrast, brightness, sharpness, shear-$x$, shear-$y$, translate-$x$, translate-$y$), the maximum steps of data augmentation: $n$, the distortion magnitude of data augmentation: $m$.
Output: the sequence of augmentation.
1: for $1, 2, \cdots, n$ do
2:    Randomly select one operation from the operation collection.
3:    Assign distortion magnitude $m$ to the data augmentation operation.
4:    return $A$ sequence of augmentation operations with length $N$.
```

Algorithm 1: RandAugment.

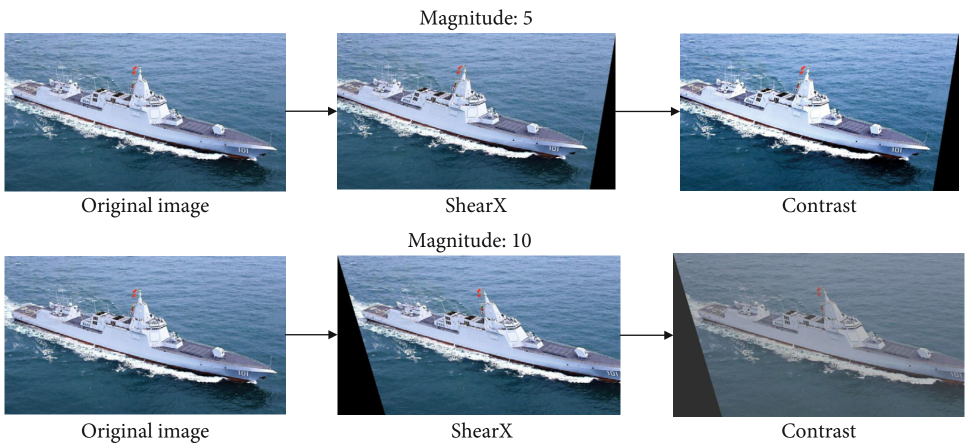


Figure 5: The effect of different distortion magnitudes in RandAugment.

the output of the convolution modules is flattened and put into a nonlinear classifier, consisting of fully connected layers, ReLUs, and a softmax function. The classifier can be used to map visual features to the probability distribution of different categories, denoted as $p_\theta(y|x)$. According to the probability distribution, the category of the image can be obtained. The parameters of the backbone network can be adjusted through the semisupervised learning algorithm to classify images more accurately.

3.2. Data Augmentation Strategy. In image classification tasks, augmentation strategies always include flipping, translation, and clipping, as illustrated in Figure 4. The augmentation strategy used in this paper is RandAugment [39], which is an improved version of AutoAugment [34]. AutoAugment is aimed at automatically finding a series of suitable image augmentation strategies from the Python Image Library (PIL) and form a final augmentation method by combining these strategies together. However, RandAugment samples strategies randomly instead of searching for them, largely improving efficiency. The set of sampling set is also from PIL. In general, it is easier to carry out RandAugment compared to AutoAugment; besides, it does not require additional labeled samples for finding suitable augmentation strategies. Algorithm 1 gives the details of RandAugment.

The effect of RandAugment relies on the number of operations n and the distortion magnitude m. Supposing that $n = 2$, the effect of different distortion magnitudes is shown in Figure 5. As we can see, the image has more changes as m gets larger.

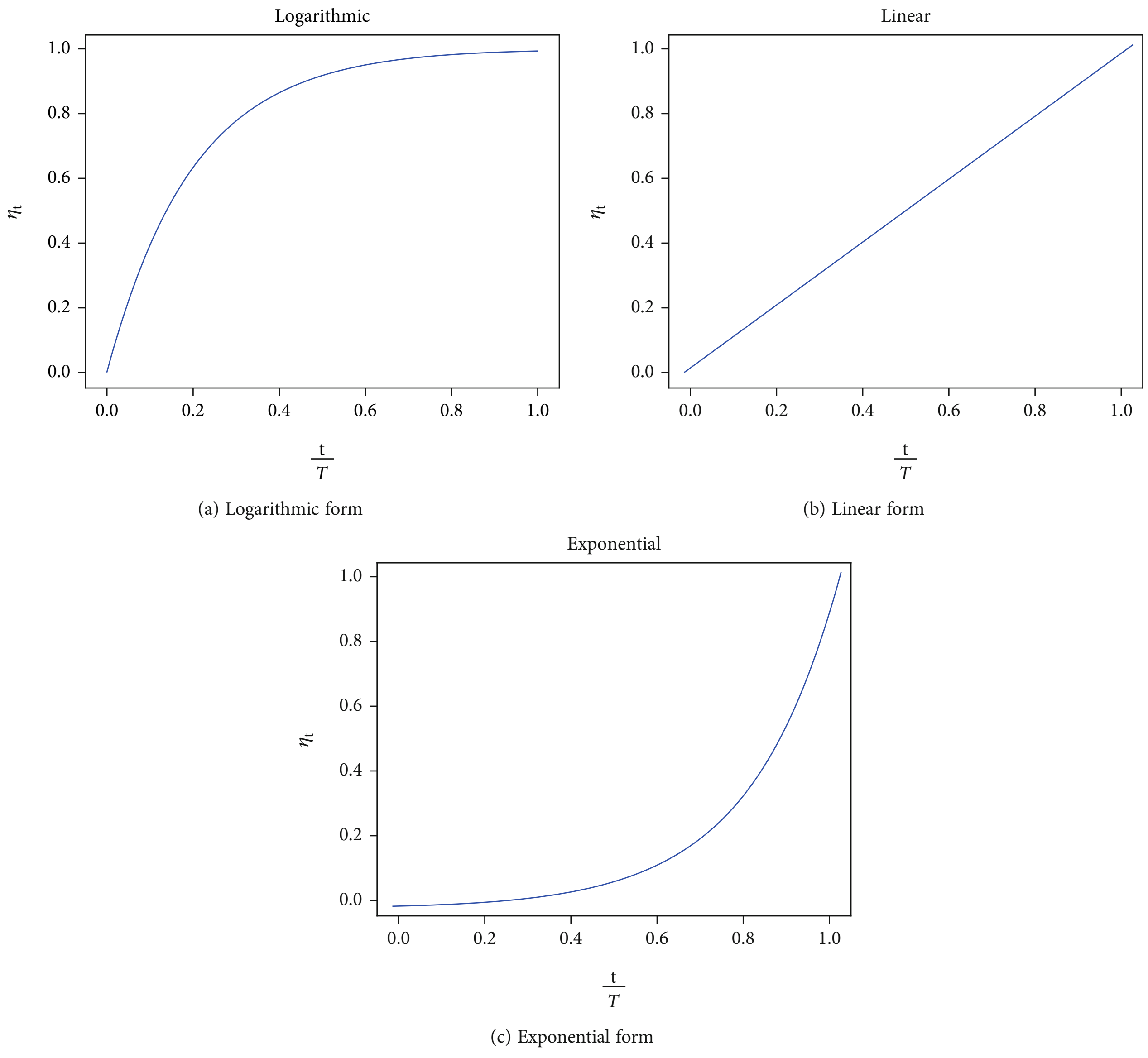
3.3. Symmetric Consistency Learning. In the original UDA algorithm, the unsupervised loss is calculated by KL-divergence, which is an asymmetric function. The measurement result will change if the sequence of data input reverses, causing the deviation of the training network weight. However, the difference between probability distributions needs to be independent of the input order. Therefore, we proposed a new measurement method called absolute log-likelihood estimation (ALE). It is symmetric and will not change when the input order changes. ALE prevents the information learned from samples from deviating. The details of ALE are described in Algorithm 2. It computes the absolute value of the difference between two probability distributions first. Then, it takes negative logarithmic on the result, and finally, probability distribution difference can be obtained. The unsupervised loss obtained by ALE is shown as

$$E_{x_2\sim p_U(x)} E_{\hat{x}\sim q(\hat{x}|x_2)}\left[\mathrm{ALE}\left(p_{\tilde{\theta}}(y|x_2)\|p_\theta(y|\hat{x})\right)\right] = \frac{1}{m_2}\sum_{i=1}^{m_2}\sum_{c=1}^{N} -\log\left(1-\left|p_{\tilde{\theta}}(y_i=c|x_i)-p_\theta(y_i=c|\hat{x}_i)\right|\right). \tag{7}$$

3.4. Additional Training Techniques. In order to achieve a better training performance, some additional training

```
**Input:** Probability distribution: $p$, another probability distribution: $q$.
**Output:** Probability distribution difference value res.
  1: Calculate the absolute difference between two probability distributions to get the intermediate variable
     $z_i \longleftarrow 1 - \mathrm{abs}(p(x_i) - q(x_i))$.
  2: Take the negative logarithm and sum them up to get the result res $\longleftarrow -\sum_i \log (z_i)$.
  3: **return** res.
```

Algorithm 2: Absolute log-likelihood estimation.



(a) Logarithmic form (b) Linear form (c) Exponential form

Figure 6: Three forms to increase the threshold η_t.

techniques are adopted in the training process and the loss function calculation.

3.4.1. Confidence Mask. In the training process, the probability distributions of unlabeled data x and unlabeled augmented data $\hat{x}$ are used to calculate the unsupervised loss. However, not all unlabeled samples in the minibatch are suitable for loss calculation. If the probability distribution predicted by the backbone network is even, the network model will be uncertain about the classification result. In that case, the unlabeled sample hardly contributes to the network training and could cause an adverse effect on training. Therefore, we set a confidence threshold β to keep such unlabeled samples from the loss calculation. The unlabeled samples will be selected to calculate the loss value only when the maximum value of its probability distribution is greater than β. For instance, in a three-classification problem, the predicted probability distribution is $[0.6, 0.3, 0.1]$. If the threshold β is more than 0.6, the unlabeled sample will be discarded. The confidence threshold β can be set to different values according to classification tasks and datasets.

3.4.2. Predicted Value Sharpening. In order to make differences between probabilities more discernible, a softmax thermal parameter τ is added to the probability distribution of the unlabeled data [40, 41], denoted as

$$p_\theta^{\text{sharp}}(y|x) = \frac{\exp\left(z_y/\tau\right)}{\sum_{y'} \exp\left(z_{y'}/\tau\right)}, \tag{8}$$

where z_y represents the output of the last fully connected layer in the classifier and will be input to the softmax function. z_y corresponds to the category label y. Adding the softmax thermal parameter sharpens the predicted probability values and accelerates the training process.

Taken together, the unsupervised loss function in Equation (7) can be redefined by combining the confidence mask and the predicted value sharpening technologies, shown as

$$\begin{aligned} &E_{x_2\sim p_U(x)}E_{\hat{x}\sim q(\hat{x}|x_2)}\left[\text{ALE}\left(p_{\tilde{\theta}}(y|x_2)\|p_\theta(y|\hat{x})\right)\right] \\ &\quad= \frac{1}{m_2}\sum_{i=1}^{m_2}\mathbf{1}\left(\max_{y'} p_{\tilde{\theta}}\left(y'|x_i\right) > \beta\right)\sum_{c=1}^{N} \\ &\quad\quad -\log\left(1-\left|p_{\tilde{\theta}}^{\text{sharp}}(y_i=c|x_i)-p_\theta(y_i=c|\hat{x}_i)\right|\right), \end{aligned} \tag{9}$$

where $\mathbf{1}(\max_{y'} p_{\tilde{\theta}}(y'|x_i) > \beta) = 1$ if $\max_{y'} p_{\tilde{\theta}}(y'|x_i) > \beta$ is true; otherwise, $\mathbf{1}(\max_{y'} p_{\tilde{\theta}}(y'|x_i) > \beta) = 0$.

3.4.3. Training Signal Annealing. In the semisupervised learning framework, labeled data is far more than unlabeled data. It leads to the situation that the network has overfitted labeled samples while it is still underfitting unlabeled samples. To tackle the problem, training signal annealing (TSA) is used to gradually release the knowledge of labeled samples into the training process. For example, in the tth iteration of training, the backbone network receives the labeled sample x and outputs the probability distribution $p_\theta(y^*|x)$. If the maximum probability value is greater than the threshold η_t, the labeled sample will not be used when calculating the supervised loss. Moreover, the value of η_t increases as training continues, ranging from $1/n_c$ to 1. n_c is the number of categories. As a result, TSA effectively prevents the network from overfitting labeled samples during the training process.

Table 1: Symbol table.

Symbol	Description
$q(\hat{x}|x)$	Data augmentation operation.
θ	Parameter set of backbone network M.
$p_\theta(y|x_1)$	Probability distribution of labeled samples.
$p_{\tilde{\theta}}(y|x_2)$	Probability distribution of unlabeled samples.
$p_L(x)$, $p_U(x)$	Data distributions of labeled samples and unlabeled samples.
n_c	Number of categories.
n_S	Number of labeled samples in the minibatch.
n_U	Number of unlabeled samples in the minibatch.
n	The maximum steps of data augmentation in RandAugment.
m	Distortion magnitude of data augmentation in RandAugment.
β	Threshold of the confidence mask.
τ	Softmax thermal parameter in the predicted value sharpening.
η_t	Threshold of the training signal annealing.

Figure 6 illustrates three strategies to increase η_t, including (a) logarithmic, (b) linear, and (c) exponential forms. And their functions are denoted as

$$\begin{aligned} \eta_t &= \left(1-\exp\left(-\frac{t}{T}*5\right)\right)*\left(1-\frac{1}{N}\right)+\frac{1}{N}(a), \\ \eta_t &= \frac{t}{T}*\left(1-\frac{1}{N}\right)+\frac{1}{N}(b), \\ \eta_t &= \exp\left(\left(\frac{t}{T}-1\right)*5\right)*\left(1-\frac{1}{N}\right)+\frac{1}{N}(c), \end{aligned} \tag{10}$$

where T is the maximum number of training iterations.

Table 1 lists the major symbols used in this paper.

4. Experiments

In this section, experiments are carried out to evaluate the performance of the proposed method. First, we introduce the experiment settings and preprocess the dataset. Then, we test the proposed method using the OpenSARShip dataset [42] and analyze the testing results.

4.1. Experiment Settings and Dataset Preprocessing. We used the OpenSARShip dataset for training and testing in this paper, which contains labeled and unlabeled SAR images of different ships. OpenSARShip is organized into different folders corresponding to different scenes. In each folder, there are four formats of ship images, including "Patch," "Patch_Uint8," "Patch_RGB," and "Patch_Cal," as shown in Figure 7. In this paper, we chose the visualized 8-bit gray images ("Patch_Uint8") as the original images for training the network. Moreover, we selected images of bulk carriers,

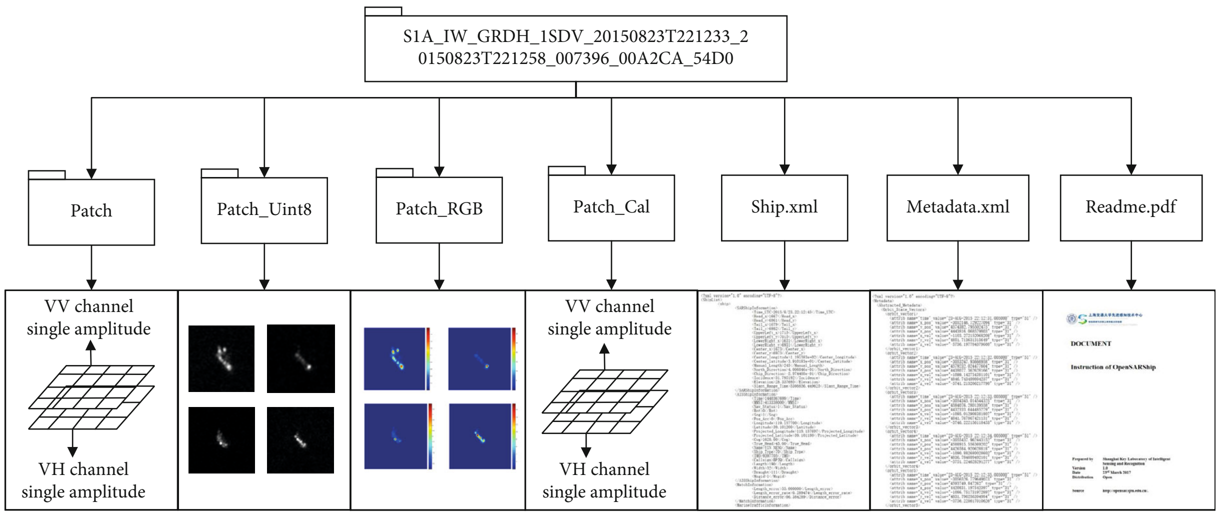

S1A_IW_GRDH_1SDV_20150823T221233_20150823T221258_007396_00A2CA_54D0
Patch
Patch_Uint8
Patch_RGB
Patch_Cal
Ship.xml
Metadata.xml
Readme.pdf
VV channel single amplitude
VH channel single amplitude
VV channel single amplitude
VH channel single amplitude

Figure 7: OpenSARShip dataset file structure.

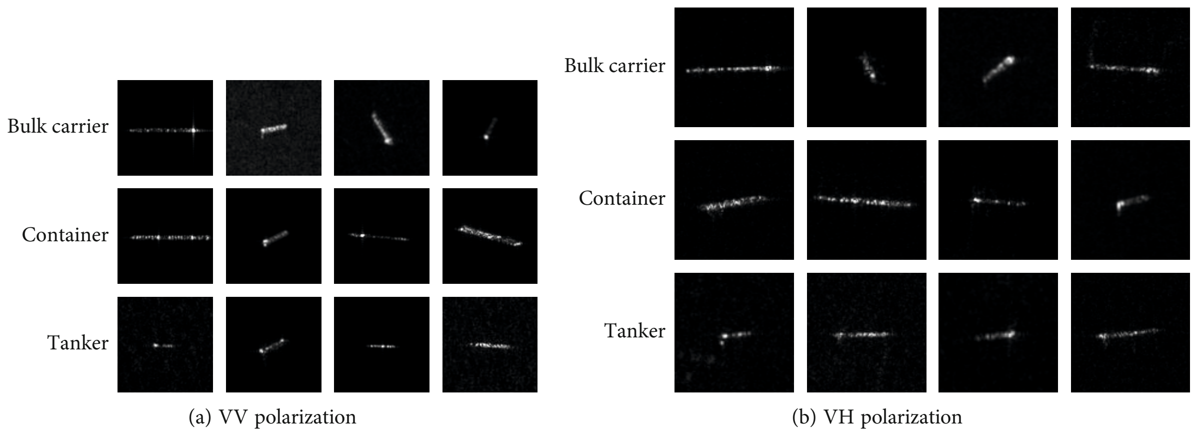


Figure 8: Examples of SAR images based on VV and VH polarizations.

containers, and tankers to formulate a three-classification problem, and each SAR image has two types of polarization (VV and VH), as shown in Figure 8.

We preprocessed the dataset before training the classification model. For labeled data, first of all, we collected 1740 chips of tankers, 1582 chips of containers, and 2298 chips of bulk carriers from the OpenSARShip dataset. Second, we chose images that are in the tanker set and larger than 35×35 pixels to build a new subset of tankers. Similarly, images that are in the container set and larger than 70×70 pixels are collected as a subset of containers, and images that are in the bulk carrier set and larger than 75×75 pixels are retained as a subset of bulk carriers. Thus, we kept 1450 chips of tankers, 1410 chips of containers, and 1442 chips of bulk carriers, which will be used for training and testing. Third, we resized these images to 100×100 pixels and divided images in each class into three parts, consisting of K samples for training, 200 samples for validation, and the rest for testing. For unlabeled data, we collected them from the remaining images in the OpenSARShip dataset and discarded their labels. There are 3000 unlabeled images in total, and all of them are also resized to 100×100 pixels. In addition, all images in the unlabeled dataset are different from images in the labeled dataset.

CNN, UDA-KL, and UDA-ALE are implemented in this section. CNN is the backbone network. UDA-KL is based on the UDA algorithm and uses KL-divergence to calculate unsupervised loss, whereas UDA-ALE uses the proposed ALE function to obtain unsupervised loss. The validation is carried out every 500 iterations. If the accuracy of the validation set no longer increases and even begins to decline, the training process terminates. During the validation and testing processes, the classification model receives samples and outputs the prediction result, denoted as

$$\hat{y}_i = \arg \max_c p_\theta(y_i = c|x_i). \tag{11}$$

The experiment parameters are listed in Table 2.

4.2. Results and Discussion. During the training process, we set up different situations based on the number of samples in each class, denoted as K. Images are sampled at random,

Table 2: Training parameters.

Parameter	Value
Number of labeled samples in a minibatch	8
Number of unlabeled samples in a minibatch	32
Initial learning rate	5×10^{-5}, being halved every 5×10^{3} iterations
Weight of unsupervised loss λ	1
Confidence threshold β	0.5
Softmax thermal parameter τ	0.5
TSA strategy	Logarithmic form
Maximum number of iterations	25000

Table 3: Classification accuracy with different K values.

Number of training images in each class (K)	CNN	UDA-KL	UDA-ALE
50	62.75%	72.93%	73.75%
100	68.52%	75.42%	76.25%
200	73.68%	77.51%	79.25%
400	78.92%	81.22%	82.33%
600	81.44%	82.36%	82.92%
800	83.15%	83.74%	84.08%

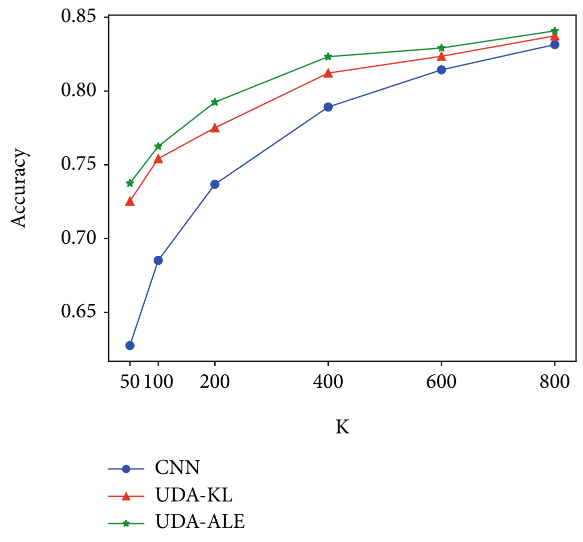


Figure 9: The diagram of classification accuracy.

and 50 independent experiments are carried out in each situation. The results are shown in Table 3 and Figure 9.

We can see that the proposed algorithm (UDA-ALE) has higher classification accuracy than UDA-KL and CNN in all situations. Compared to CNN, UDA-KL utilizes both labeled and unlabeled samples and learns more potential information during the training process; therefore, it reaches a higher accuracy rate. Furthermore, UDA-ALE uses the ALE function, which is symmetric, to calculate unsupervised loss instead of using KL-divergence, which is asymmetric. Therefore, compared to UDA-KL, UDA-ALE has better robustness to the input order of probability distributions and classifies SAR images more accurately. In general, symmetric consistency learning is more suitable for solving SAR image classification problems.

5. Conclusion

This paper proposed a semisupervised learning-based method to solve SAR image classification with few labeled data. The method uses labeled and unlabeled data for model training, which makes most the of information in unlabeled samples and avoids the overfitting problem due to the lack of labeled samples. In addition, a novel symmetric function is designed to calculate unsupervised loss, which changes the measurement mode of the difference between probability distributions in the original UDA framework. The experiment results show that the original UDA performs better than the backbone network because it makes the best use of information in unlabeled images. Besides, the proposed method reaches a much higher classification accuracy than the original UDA because symmetric consistency learning makes the training process more robust and steady. In the future, we will concentrate on minimizing the size of the proposed model to make it suitable for resource-constraint IoT devices.

Data Availability

The data used to support this study have not been made available.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Yibing Li and Sitong Zhang are coprimary authors.

Acknowledgments

This paper is funded by the National Defense Science and Technology Key Laboratory Program (J2322010) and Heilongjiang Touyan Innovation Team Program.

References

[1] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 144–153, Dallas, Texas, USA, 2019.

[2] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 968–979, 2020.

[3] Y. Zhou, H. Wang, F. Xu, and Y.-Q. Jin, "Polarimetric SAR image classification using deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 12, pp. 1935–1939, 2016.

[4] J. Geng, J. Fan, H. Wang, X. Ma, B. Li, and F. Chen, "High-resolution SAR image classification via deep convolutional autoencoders," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 11, pp. 2351–2355, 2015.

[5] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[6] R. Shang, J. He, J. Wang, K. Xu, L. Jiao, and R. Stolkin, "Dense connection and depthwise separable convolution based CNN for polarimetric SAR image classification," *Knowledge-Based Systems*, vol. 194, article 105542, 2020.

[7] H. Zhu, W. Wang, and R. Leung, "SAR target classification based on radar image luminance analysis by deep learning," *IEEE Sensors Letters*, vol. 4, no. 3, pp. 1–4, 2020.

[8] Z. Cai, X. Zheng, J. Wang, and Z. He, "Private data trading towards range counting queries in internet of things," *IEEE Transactions on Mobile Computing*, p. 1, 2022.

[9] J. Ni, F. Zhang, Q. Yin, Y. Zhou, H.-C. Li, and W. Hong, "Random neighbor pixel-block-based deep recurrent learning for polarimetric SAR image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 9, pp. 7557–7569, 2021.

[10] T. Tai and M. Toda, "Adapting intra-class variations for SAR image classification," in *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 2653–2657, Anchorage, Alaska, USA, 2021.

[11] G. Chen, L. Wang, and M. Kamruzzaman, "Spectral classification of ecological spatial polarization SAR image based on target decomposition algorithm and machine learning," *Neural Computing and Applications*, vol. 32, no. 10, pp. 5449–5460, 2020.

[12] A. Zhang, X. Yang, S. Fang, and J. Ai, "Region level SAR image classification using deep features and spatial constraints," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 163, pp. 36–48, 2020.

[13] Y. Zhang, F. Liu, L. Jiao, S. Yang, L. Li, and M. Yang, "Discriminative sketch topic model with structural constraint for SAR image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 5730–5745, 2020.

[14] Z. Sun, J. Li, P. Liu, W. Cao, T. Yu, and X. Gu, "SAR image classification using greedy hierarchical learning with unsupervised stacked CAEs," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, pp. 5721–5739, 2021.

[15] Z. Ren, B. Hou, Q. Wu, Z. Wen, and L. Jiao, "A distribution and structure match generative adversarial network for SAR image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 6, pp. 3864–3880, 2020.

[16] J. Geng, X. Deng, X. Ma, and W. Jiang, "Transfer learning for SAR image classification via deep joint distribution adaptation networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 8, pp. 5377–5392, 2020.

[17] L. Li, L. Ma, L. Jiao, F. Liu, Q. Sun, and J. Zhao, "Complex contourlet-CNN for polarimetric SAR image classification," *Pattern Recognition*, vol. 100, article 107110, 2020.

[18] K. Dasari, A. Lokam, and P. Jayasri, "A study on utilization of polarimetric SAR data in planning a smart city," *Procedia Computer Science*, vol. 93, pp. 967–974, 2016.

[19] H. Lang, J. Zhang, X. Zhang, and J. Meng, "Ship classification in SAR image by joint feature and classifier selection," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 2, pp. 212–216, 2016.

[20] H. Lang, S. Wu, and Y. Xu, "Ship classification in SAR images improved by AIS knowledge transfer," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 3, pp. 439–443, 2018.

[21] M. Jiang, X. Yang, Z. Dong, S. Fang, and J. Meng, "Ship classification based on superstructure scattering features in SAR images," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 5, pp. 616–620, 2016.

[22] W. T. Chen, K. F. Ji, X. W. Xing, H. X. Zou, and H. Sun, "Ship recognition in high resolution SAR imagery based on feature selection," in *2012 International Conference on Computer Vision in Remote Sensing*, pp. 301–305, Xiamen, China, 2012.

[23] D. A. Morgan, "Deep convolutional neural networks for ATR from SAR imagery," in *Algorithms for Synthetic Aperture Radar Imagery XXII*, vol. 9475, article 94750F, International Society for Optics and Photonics, 2015.

[24] Y. Li, X. Li, Q. Sun, and Q. Dong, "SAR image classification using CNN embeddings and metric learning," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2020.

[25] F. Zhang, Y. Wang, J. Ni, Y. Zhou, and W. Hu, "SAR target small sample recognition based on cnn cascaded features and AdaBoost rotation forest," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 6, pp. 1008–1012, 2020.

[26] S. Chen, H. Wang, F. Xu, and Y.-Q. Jin, "Target classification using the deep convolutional networks for SAR images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4806–4817, 2016.

[27] K. Chen, Z. Pan, Z. Huang, Y. Hu, and C. Ding, "Learning from reliable unlabeled samples for semi-supervised SAR ATR," *IEEE Geoscience and Remote Sensing Letters*, vol. 19, pp. 1–5, 2022.

[28] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision*, pp. 818–833, Springer, 2014.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[30] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, Boston, MA, USA, 2015.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, Las Vegas, NV, USA, 2016.

[32] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[33] A. W. Yu, D. Dohan, M.-T. Luong et al., "QANet: combining local convolution with global self-attention for reading comprehension," 2018, https://arxiv.org/abs/1804.09541.

[34] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAugment: learning augmentation policies from data," 2018, https://arxiv.org/abs/1805.09501.

[35] A. Hannun, C. Case, J. Casper et al., "Deep Speech: scaling up end-to-end speech recognition," 2014, https://arxiv.org/abs/1412.5567.

[36] D. S. Park, W. Chan, Y. Zhang et al., "SpecAugment: a simple data augmentation method for automatic speech recognition," 2019, https://arxiv.org/abs/1904.08779.

[37] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," 2019, https://arxiv.org/abs/1904.12848.

[38] T. Van Erven and P. Harremos, "Rényi divergence and Kullback-Leibler divergence," *IEEE Transactions on Information Theory*, vol. 60, no. 7, pp. 3797–3820, 2014.

[39] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "RandAugment: practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 702-703, Seattle, Washington, USA, 2020.

[40] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," *Advances in Neural Information Processing Systems*, vol. 367, pp. 281–296, 2005.

[41] T. Miyato, S. I. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1979–1993, 2019.

[42] J. Zhao, Z. Zhang, W. Yao, M. Datcu, H. Xiong, and W. Yu, "OpenSARUrban: a sentinel-1 SAR image dataset for urban interpretation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 187–203, 2020.