SCL-MLNet: Boosting Few-Shot Remote Sensing Scene Classification via Self-Supervised Contrastive Learning

Xiaomin Li, Daqian Shi, Xiaolei Diao, and Hao Xu

***Abstract*— Few-shot classification aims at recognizing novel categories from low data regimes based on prior knowledge. However, the existing methods for few-shot scene classification have limitations on using few annotated data and do not fully consider the intra-class samples with classification targets in different sizes, which lead to poor feature representation. To address these problems, this study introduces an end-to-end framework called self-supervised contrastive learning-based metric learning network (SCL-MLNet) for few-shot remote sensing (RS) scene classification. On one hand, we weave self-supervised contrastive learning into few-shot classification algorithms through multi-task learning, enabling feature extractors to learn representative image features from few annotated samples. Moreover, we devise a new loss function to train the proposed model end-to-end and speed up the convergence of the model. On the other hand, considering the differences between intra-class samples, we introduce a novel attention module embedded in the feature extractor to fuse multi-scale spatial features from the classification targets in different sizes. In our experiments, SCL-MLNet is evaluated on three public benchmark datasets. The results demonstrate that SCL-MLNet achieves state-of-the-art performance for few-shot remote sensing scene classification.**

***Index Terms*— Feature representation, few-shot learning, remote sensing (RS) scene classification, self-supervised contrastive learning (SCL).**

I. Introduction

REMOTE sensing scene classification has aroused considerable interest among researchers due to its wide and valuable applications, such as disaster monitoring, geological prospecting, and road planning [1]–[4]. Deep learning is widely used for remote sensing (RS) scene classification [5]–[7], which extracts feature representation with convolutional neural networks (CNNs) [1]. Earlier CNNs mainly include AlexNet [8], VGGNet [9], GoogleNet [10], ResNet [11], and improved networks [12], all of which have achieved tremendous success in remote sensing scene classification. However, the success of these methods relies on a large amount of manually annotated data for training. When facing with limited annotated samples, CNNs tend to overfit during training, resulting in unsatisfactory generalization performance [13]. Moreover, as the number of higher resolution scene images increases, it is time-consuming and expensive to annotate massive images in real situation, especially in some specific fields, such as remote sensing scene images, medical images, and ancient images due to the lack of professional background knowledge [14], [15]. Furthermore, humans can learn new visual concepts from only a few samples and generalize with no difficulty to unseen data [14], [15], but a deep learning model trained with limited samples severely lacks this ability [2], [15], [16]. Few-shot learning endows artificial intelligence systems with a similar capacity to quickly learn from prior knowledge, even though there are few annotated data [13], [16], [17]. Most of the extant few-shot learning methods are based on meta-learning [18], the goal of which is to develop a classification model that is capable of efficiently classifying a series of classes with few available examples of each class (e.g., only one or five samples per class) [16], [14], [19]. It divides an entire few-shot classification task into several small tasks by subsampling classes and data.

In the early stage of few-shot learning, Bengio *et al.* [20], Fei-Fei *et al.* [17], and Fei-Fei *et al.* [21] conducted some important work on classification for only a few images. Gradually, few-shot learning methods based on meta-learning emerged [22]–[28] and many of them are used in remote sensing scene classification [2], [29]. As shown in Fig. 1, the images of the two lines have intra-class samples with classification targets in different sizes because of the different distances and angles between scenes and remote sensing satellites. It may result in the entire target object or only portion of it is covered in the same size of a receptive field, preventing us from obtaining a complete representation. Nevertheless, most existing studies [2], [30] use a unified convolution kernel to extract image features, which can result in poor quality feature representation due to different coverage areas.

In addition, the existing methods [2], [29], [30] for few-shot scene classification have limitations in terms of using few annotated data, which results in poor representations. Self-

Manuscript received April 4, 2021; revised July 8, 2021 and August 23, 2021; accepted August 25, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 62077027, in part by the Ministry of Science and Technology of the People's Republic of China under Grant 2018YFC2002500, in part by the Jilin Province Development and Reform Commission, China under Grant 2019C053-1, in part by the Education Department of Jilin Province, China under Grant JJKH20200993K, in part by the Department of Science and Technology of Jilin Province, China under Grant 20200801002GH, and in part by the European Union's Horizon 2020 Future and Emerging Technologies (FET) Proactive project "WeNet-The Internet of us" under Grant 823783. *(Corresponding author: Hao Xu.)*

Xiaomin Li and Hao Xu are with the College of Computer Science and Technology, Jilin University, Changchun 130012, China (e-mail: xuhao@jlu.edu.cn).

Daqian Shi and Xiaolei Diao are with the Department of Information Engineering and Computer Science, University of Trento, 38122 Trento, Italy.

Digital Object Identifier 10.1109/TGRS.2021.3109268

Fig. 1. Remote sensing scene images from the NWPU-RESISC45 dataset. The top line of images has large classification targets and the bottom line has small classification targets.

supervised contrastive learning (SCL) provides a promising solution for making the most of a few annotated data [31]. As a subset of unsupervised learning methods, SCL is used to learn image feature representations by leveraging the structural information generated from the data as the supervisory signal [32], which greatly alleviates the problem of high annotation cost [33]–[35]. Besides, unannotated samples can be used in SCL to extend training data so that the feature extracting ability of the framework will be enhanced without additional requirements.

Based on the above background, we propose an end-to-end multi-task framework for few-shot scene classification through SCL, called SCL-based metric learning network (SCL-MLNet). First, a feature extractor is trained using SCL as an auxiliary task in the few-shot classification pipeline to get detailed image features even though there are few annotated samples. The core idea of SCL is to train networks to maximum agreement of different views of the same images while minimizing the agreement of views from different images [32], [34]. In addition, we devise a new loss function to train our framework end-to-end and speed up convergence, which can balance SCL loss and few-shot classification loss. Second, we introduce a novel attention module embedded in the feature extractor, which can fuse multi-scale spatial features from classification targets in different sizes. Our attention module consists of a channel attention module and a spatial attention module. The widely used squeeze-and-excitation convolution (SE-Conv) [36] is used as our channel attention module. Remote sensing scene images are captured from different distances and angles, resulting in objects of the same category having varying sizes and making deciding convolution kernel size in advance difficult. To fuse multi-scale features from large classification targets and small classification targets, the spatial attention module uses convolution kernels of different sizes.

In summary, our main contributions are as follows.

1) We propose an end-to-end multi-task framework named SCL-MLNet for few-shot remote sensing scene classification through SCL module. Different from the existing few-shot methods, SCL-MLNet can use both the information in the data and the labels as the supervisory signal.
2) Based on SCL-MLNet, we devise a novel loss function that can balance SCL loss and few-shot classification loss by endowing them appropriate weight coefficients.
3) Considering the case that the intra-class classification targets are in different sizes, we introduce a novel attention module to fuse multi-scale features from classification targets in different sizes to enrich visual representations, which is plug-and-play and extensible.
4) We achieve state-of-the-art performance compared with representative few-shot learning methods on three widely used remote sensing scene classification datasets.

The remainder of this article is organized as follows. In Section II, some crucial work on few-shot learning and remote sensing scene classification is introduced. Section III mainly elaborates on the overall architecture of SCL-MLNet and our embedding module. Section IV displays the experimental results on three publicly available datasets and analyzes the results. Finally, conclusion and summaries are presented in Section V.

II. Related Works

In this section, we mainly introduce some works related to remote sensing scene classification, few-shot learning, and meta-learning.

A. Remote sensing Scene Classification

Remote sensing scene classification has aroused considerable interest recently, most of which are based on deep learning [37]. For example, Xie *et al.* [38] introduced a scale-free CNN (SF-CNN) to prevent information discard and poor classification performance caused by a resizing process when fine-tuning pretrained CNNs. Lu *et al.* [39] proposed an end-to-end CNN module (FACNN) to leverage the semantically annotated information rather than using only unsupervised feature encoding methods to aggregate intermediate features. Xu *et al.* [40] presented a two-stream feature aggregation deep neural network combining discriminative features and general features to learn detailed features. Sun *et al.* [1] proposed an end-to-end gated bidirectional network to aggregate hierarchical features and eliminate interference information. Xu *et al.* [41] investigated an adversarial training strategy to increase the resistibility of deep models and prevent adversarial examples from being misclassified. All the work mentioned above proposed a novel deep learning model to deal with different problems in remote sensing scene classification. However, deep neural networks need to be trained on large volumes of annotated data; otherwise, they will suffer from poor generalization performance. To tackle this problem, we adopt few-shot learning methods that can train deep models with a few annotated data and easily adapt to novel classes by dividing an entire classification task into several subtasks. In addition, we also weave self-supervised contrastive learning into the model to enrich feature representation.

B. Few-Shot Learning and Meta-Learning

Few-shot learning focuses on optimization methods that can efficiently learn feature representation from low data regimes [33]. There are many few-shot learning methods,

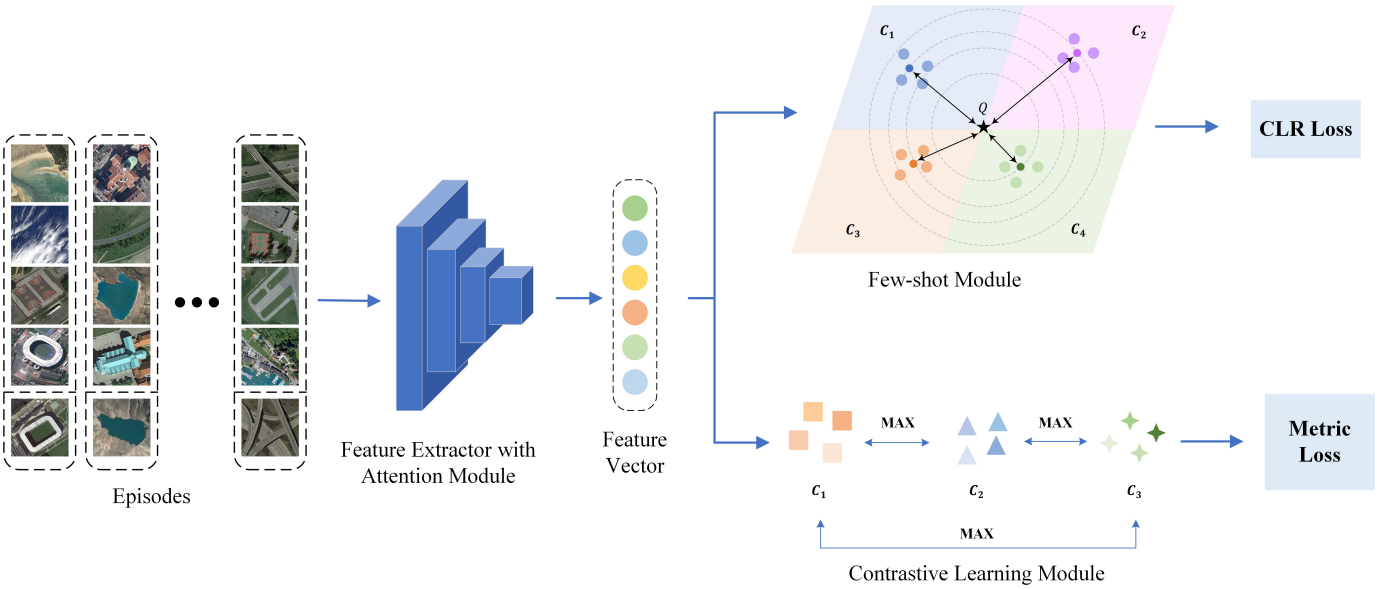

Fig. 2. Illustration of the proposed SCL-MLNet framework. SCL-MLNet is trained in a meta-learning manner. In each episode, N classes are randomly selected from the training/test set, and then we select K images per class as the support set and Q images per class as the query set. This method learns the embedding of the labeled examples (the support set), which can be used to predict classes for the unlabeled points (the query set) by distance in the embedding space.

such as metric-learning-based approaches by learning a distance metric between a query (i.e., support images, query images), including matching networks (MatchingNet) [22], prototypical networks (ProtoNet) [23], and relation networks (RelationNet) [24]. MatchingNet aims at achieving rapid learning using recent advances in attention and memory. ProtoNet learns a prototype representation for each class and classifies an embedded query sample by finding the nearest class prototype. Different from the prior metric-based few-shot approaches, most of these rely on a manually defined similarity metric for classification. RelationNet is able to classify images through computing the relation scores produced by a neural network. In addition, there are also some gradient-descent-based approaches, such as model-agnostic meta-learning (MAML) [26], Meta long short-term memory (LSTM) [27], and Meta-stochastic gradient descent (SGD) [28], which can adapt a model to a new few-shot classification task rapidly [33]. The key idea of MAML is to learn a set of initial parameters such that when meeting a new task, the model has maximal performance after the parameters have been updated through one or more gradient steps computed with a small amount of data from that new task. Meta LSTM trains an LSTM-based meta-learner to learn an update rule for training a neural network, which can capture both short-term knowledge within a task and long-term knowledge common among all the tasks. Meta-SGD is an SGD-like meta-learner, which is easy to train and can adapt any differentiable learner in just one step.

The specific advantages of few-shot learning also attract many researchers to apply it to remote sensing scene classification tasks. Li *et al.* [2] proposed a method called discriminative learning of adaptive match network (DLA-MatchNet) with a new attention model, which can automatically discover discriminative regions to extract more discriminative features. When using metric learning for classification, DLA-MatchNet used a nonlinear network to measure the similarity between query images and support sets. Li *et al.* [30] presented a framework RS-MetaNet and a novel loss function named balance loss. RS-MetaNet aimed at solving problems that the existing methods were performed in a sample-level manner and led to overfit easily to individual samples. Zhai *et al.* [29] devised a lifelong few-shot learning model to make use of limited annotated data and deal with the disparities of the datasets from different research institutions. Nevertheless, the above-mentioned approaches ignore the point that although samples come from the same category, different samples have classification targets in different sizes, which results in poor feature representations. Thus, we need to improve the existing few-shot classification framework by devising a novel attention module to get better feature representations.

III. Methodology

A. Overview

To learn more detailed feature representations from a few annotated data and fuse multi-scale features from classification targets in different sizes, we devise a framework called SCL-MLNet. As shown in Fig. 2, we train our model in a meta-training manner by learning a metric space on several tasks sampled from a task set. Then, a feature extractor with our novel attention module is used to generate the feature vectors, which are used as inputs to the few-shot classification module and contrastive learning module. We adopt the prototypical network as the few-shot classification method and apply our contrastive learning module in the few-shot classification pipeline. Through the integration of these two tasks, we combine the contrastive learning loss with the classification loss to enhance the feature representation ability of the feature extractor.

B. Problems Definition and Notations

In this work, we use the splits introduced by [27] to divide each dataset $\mathcal{D}$ with a total of N^* classes into a training set D_{tn}, validation set D_{val}, and test set D_{ts}. The above splits have their own label space and disjoin with each other, that is, $L_{\text{tn}} \cap L_{\text{val}} = \emptyset$, $L_{\text{tn}} \cap L_{\text{ts}} = \emptyset$, $L_{\text{val}} \cap L_{\text{ts}} = \emptyset$, and $|L_{\text{total}}| = |L_{\text{tn}} \cup L_{\text{val}} \cup L_{\text{ts}}|$. In addition, three splits play different roles in the few-shot classification task, i.e., D_{tn} aims at training a model, D_{val} is used to adjust parameters and evaluate the performance of the model preliminarily, and D_{ts} can assess the model's generalization ability to unseen data.

To use the episode-based manner of meta-learning for training and testing, a few-shot learning task must be divided into several subtasks. Each subtask uses the data from the support set to train a model and use the data in the query set for testing. The support set D_s contains C different categories, each of which has K samples, and this approach is called C-way K-shot, namely, $D_s = \{(x_i, y_i)\}$ and $|D_s| = C \times K$. Similarly, there are Q samples of each class in the query set, which meets $D_q = \{(x_j, y_j)\}$ and $|D_q| = C \times Q$. Moreover, no matter how many categories are in the test set, each meta-test task is set as C-way classification, and the number of categories we applied in our experiments will depend on the realistic needs. In fact, we adopt five-way assumption in our experiments, which is widely adopted by many methods [2], [22]–[30]. When applying our method in practice, we can modify the number of categories according to realistic task.

C. Feature Extractor With Attention Module

The overall feature extractor contains a backbone with three convolutional blocks and two attention modules, which are used to learn detailed and generic feature representations, as shown in Fig. 3(a). We choose the shallow network as the backbone rather than popular networks, such as VGGNet and Resnet, because the performance of few-shot classification decreases as the number of network layers increases in our experiments, and this phenomenon will be further discussed in Section IV.

As shown in Fig. 3(b), each convolutional block is composed of two similar convolutional layers, and each layer contains a 3×3 convolution, a batch normalization, a rectified linear unit (ReLU) activation nonlinearity layer, and a 2×2 max-pooling layer. Note that different convolution layers have different numbers of channels and batch normalization sizes. Formally, let x_i^* represents a random image in D_{tn} and $\boldsymbol{F}$ denotes the image feature representation learned by our feature extractor, and then the embedding mapping of x_i^* can be written as follows:

$$\boldsymbol{F} = f_\varphi\left(x_i^*; \varphi\right) \tag{1}$$

where φ are the parameters of the feature extractor.

In addition, we also introduce a novel attention module called channel attention and spatial attention module (CSAM) to fuse multi-scale feature representations from classification targets in different sizes. As the name suggests, our CSAM consists of a channel attention and a spatial attention module. As shown in Fig. 3(c), the channel attention and spatial

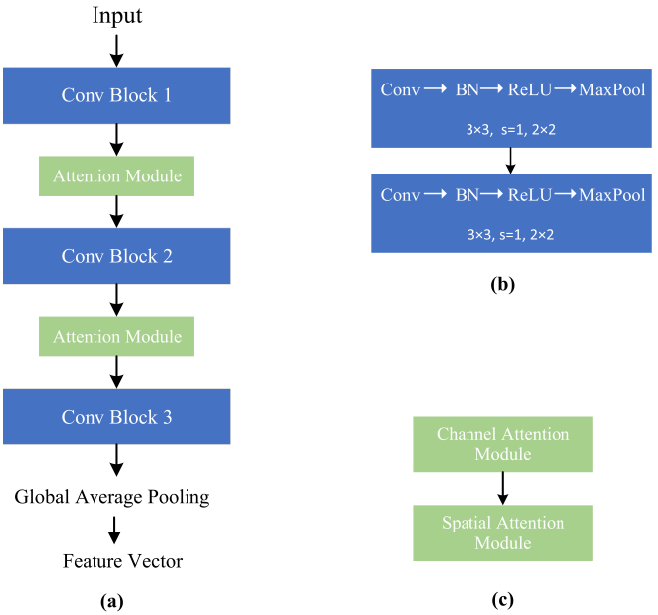

Fig. 3. Overall architecture of feature extractor. (a) Overall architecture. (b) Convolutional block with two layers. (c) Attention module consisting of CSAM.

attention modules are organized in sequence, that is, we first use the channel attention module, and then use its output as the input for the spatial attention module.

In the attention module, $\mathcal{C}$ represents the feature map, where the channel attention module focuses on, and $\mathcal{S}$ represents the feature map, where the spatial attention module focuses on. As described in Fig. 4(a), first, we can obtain the initial feature map $\text{conv}(x_i^*)$ of x_i^* by learning with the first convolution block. Then, the feature vector $\boldsymbol{Z}$ will be obtained by performing the global average pooling and flattening operation on the above feature map. Afterward, the feature vector of $\boldsymbol{Z}$ is successively fed into two fully connected layer and a Sigmoid activation function σ is used after the second fully connected layer to control the weight of each channel ranging from [0, 1]. Finally, we can obtain the feature map $\mathcal{C}$ by multiplying the channel weight matrix with the initial feature map $\text{conv}(x_i^*)$. Therefore, the mapping process is organized by the following equations:

$$\boldsymbol{Z} = \text{AvgPool}_s\left(\text{conv}\left(x_i^*\right)\right) \tag{2}$$

$$\mathcal{C} = \text{conv}\left(x_i^*\right) \cdot \sigma\left(fc_2(\text{ReLU}(fc_1(\boldsymbol{Z})))\right). \tag{3}$$

In addition, spatial attention is used to enhance feature representations based on the feature map obtained by the channel attention module. As shown in Fig. 4(b), first, we conduct an average operation on $\mathcal{C}$ along the channel axis to obtain a spatial feature descriptor $\boldsymbol{T} \in \mathbb{R}^{1 \times H \times W}$. To fuse feature representations from classification targets in different sizes, we use different convolution kernel combinations. We also conduct experiments on which combination is the best choice for our spatial attention module in Section IV. It is worth noting that convolution kernel combination is not an unchangeable setting, and it is used for chasing the best results on datasets. We can obtain two feature maps of the same size using the convolutional kernels. After that, we introduce a fully

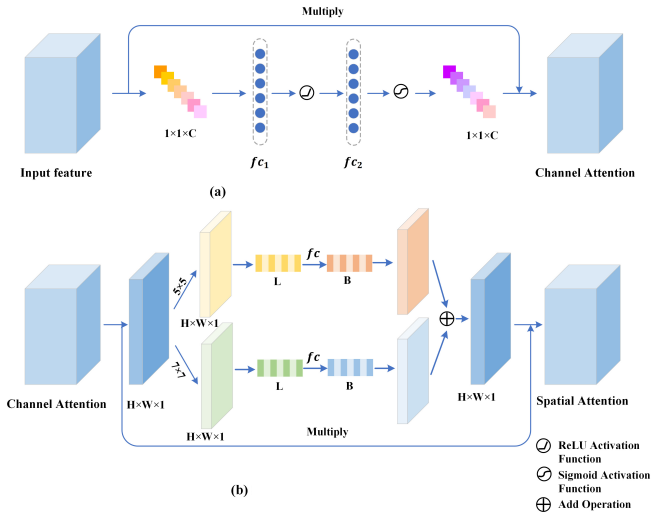

Fig. 4. Architecture of the channel attention module and spatial attention module. The number of parameters is ∼5M and the number of FLOPs is ∼1G.

connected layer into the spatial attention module to mix global features and obtain two vectors $\boldsymbol{B}$. Afterward, we restore the vectors $\boldsymbol{B}$ to the original size $H \times W \times 1$, and multi-scale feature fusion operation is performed on the above feature maps. Finally, we use the Sigmoid activation function to get the weights of each spatial position and multiply the weight matrix with $\mathcal{C}$. The feature map $\mathcal{S}$ is defined as

$$\boldsymbol{S} = \mathcal{C} \cdot \sigma\left(\left(fc\left(\text{conv}_{\text{fl}}^{i\times i}(\boldsymbol{T})\right)\right)_{\text{vec2mtx}} + \left(fc\left(\text{conv}_{\text{fl}}^{j\times j}(\boldsymbol{T})\right)\right)_{\text{vec2mtx}}\right) \quad (4)$$

where $\text{conv}_{\text{fl}}^{i\times i}$ is a $i \times i$ convolution operation on a vector, and $\text{conv}_{\text{fl}}^{j\times j}$ is a $j \times j$ convolution operation on a vector. In our experiments, i and j are set to 5 and 7, respectively. The reason will be discussed in Section IV. The option *vec2mtx* is used to restore a vector to a matrix.

In the end, the feature extractor carries out global average pooling on the output of Conv Block 3, and we obtain the final representation of the input image.

D. Applying SCL to Few-Shot Learning

We propose a novel framework weaving SCL in the first stage of few-shot learning to alleviate the dependence of deep learning methods on large amounts of labeled data. SCL is trained by reducing the distance between representations of different augmented views of the same image (positive pairs) and increasing the distance between representations of augmented views from different images (negative pairs) [42]. During this stage, the embedding module could learn discriminative features from samples. More concrete, for an input sample x_i, we can obtain two different positive views $x_{i,1}^+$ and $x_{i,2}^+$ by performing different data augmentations on it, such as random resize, random horizontal flipping, and random gray scale. Other samples are used as the negative views of x_i, which are uniformly represented as x_j^-. Next, we use positive views and negative views as input to train the feature extractor f_φ. Then, we can obtain the feature representations $\mathscr{R}$ after a nonlinear transformation.

Furthermore, the metric learning model used in this study is a prototypical network (ProtoNet). ProtoNet maps samples from the support set and query set into embeddings, and the average of embeddings in each class of the support set is regarded as the prototype of each class. Through calculating the similarity between each prototype and each query embedding, the nearest class prototype is the correct category of the query sample.

We train the feature extractor through a series of episodes during its first stage. In each episode, C classes are randomly selected from N_n classes, and N samples are randomly selected from each class. Then the support set D_* is organized based on all the $C \times N$ samples, which meet $D_* = (x^*, y^*) \subset D_{\text{tn}}$. After that, we can obtain the feature vectors of the samples in D_* using the feature extractor. We regard the average of the feature vectors from each class as the prototype of the class

$$p_j = \frac{1}{N}\sum_{i=1}^{N} f_\varphi\left(x_i^*; \varphi\right) \quad (5)$$

where i denotes the ith sample, and p_j represents the prototype of the jth class.

To classify a new sample x_q from the query set, we first obtain the feature vector $f_\varphi(x_q; \varphi)$ and calculate the Euclidean distance between the feature vector of x_q and the prototype of each class. The category of the new sample will be the same as its closest prototype. The score for each class j can be calculated as

$$S_j = \frac{\text{sim}^j\left(f_\varphi\left(x_q; \varphi\right), p_i\right)}{\sum_{k=1}^{C} \text{sim}^k\left(f_\varphi\left(x_q; \varphi\right), p_i\right)}, \quad \text{where } i \in y^* \quad (6)$$

where $\text{sim}(\cdot, \cdot)$ represents the similarity between feature vectors, which can be calculated using the Euclidean distance or cosine similarity.

E. Loss Function

Based on SCL-MLNet, we introduce a new loss function by balancing the SCL loss and few-shot classification loss, which can train our framework end-to-end. Formally, given a dataset D_S, the self-supervised loss is denoted as $L_s(\varphi; D_S)$, and φ represents the parameters of the feature extractor. The few-shot classification loss in the first stage is $L_f(\varphi; D_S)$. Thus, the total loss is as follows:

$$J(\varphi) = \min_\varphi\left(\sum_{i=1}^{n} \lambda L_f^i(\varphi; D_S) + \mu L_s^i(\varphi; D_S)\right) \quad (7)$$

where n is the number of episodes, and λ and μ are both positive values to measure the importance of few-shot loss and self-supervised loss, respectively. We will verify how different values of λ and μ impact the classification results in Section IV. Moreover, $\mathbf{D}_{\mathbf{S},\mathbf{1}}^+$ and $\mathbf{D}_{\mathbf{S},\mathbf{2}}^+$ represent two enhanced views of the positive vector sets. $\mathbf{D}_{\mathbf{S}}^+$ and $\mathbf{D}_{\mathbf{S}}^-$ represent the positive vector set and the negative vector set, respectively. Then, the SCL loss is expressed as

$$L_s^i(\varphi; D_S) = -\log\left\{\frac{\exp\left[\text{Sum}^{-1}\text{sim}\left(\mathbf{D}_{\mathbf{S},\mathbf{1}}^+, \mathbf{D}_{\mathbf{S},\mathbf{2}}^+\right)\right]}{\exp\left[\text{Sum}^{-1}\text{sim}\left(\mathbf{D}_{\mathbf{S}}^+, \mathbf{D}_{\mathbf{S}}^-\right)\right]}\right\} \quad (8)$$

TABLE I
SPLITS ON THREE DATASETS

Dataset	Training Category	Validation Category	Test Category
NWPU	Airplane, Baseball diamond, Beach, Bridge, Chaparral, Church, Cloud, Desert, Freeway, Golf Course, Harbor, Island, Lake, Meadow, Mobile Home Park, Mountain, Palace, Railway, Rectangular Farmland, Roundabout, Sea Ice, Ship, Sparse Residential, Stadium, Wetland	Industrial Area, Overpass, Railway Station, Runway, Snowberg, Storage Tank, Tennis Court, Terrace, Thermal Power Station	Airport, Basketball Court, Circular Farmland, Dense Residential, Forest, Ground Track Field, Intersection, Medium Residential, Parking Lot, River
UCM	Agricultural, Baseball Diamond, Buildings, Chaparral, Dense Residential, Freeway, Harbor, Medium Residential, Overpass, Parking Lot	Airplane, Forest, Intersection, Runway, Storage Tanks	Beach, Golf Course, Mobile Home Park, River, Sparse Residential, Tennis Court
AID	Bare Land, Baseball Field, Bridge, Church, Commercial, Dense Residential, Farm land, Forest, Industrial, Medium Residential, Mountain, Parking, Pond, Port, Resort, School	Beach, Center, Meadow, Park, Playground, Square, Stadium	Airport, Desert, Railway Station, River, Sparse Residential, Storage Tanks, Viaduct

where Sum^{-1} denotes the summation operation along the last dimension. Few-shot classification uses cross-entropy loss as its loss function, and we use Adam [43] optimizer to optimize our proposed framework SCL-MLNet.

IV. EXPERIMENTS AND ANALYSIS

In this section, we mainly introduce the experiments and analyze the results. First, three public datasets and parameters setting are displayed. Afterward, we compare our few-shot learning framework SCL-MLNet with other classical few-shot learning methods and choose proper weighting coefficients for the SCL loss and few-shot classification loss. We also explore the influence of different attention modules used in the feature extractor and display an attention visual heat map. Finally, we conduct ablation experiments to show the performance of our proposed framework.

We use PyTorch [44] development framework to implement SCL-MLNet, which is deployed on two NVIDIA GeForce RTX 3090 GPU, and the version of CUDA is 11.0.

A. Dataset Description

To verify the effectiveness of our framework, we conduct experiments on three commonly used publicly available remote sensing scene classification datasets, namely, North-western Polytechnical University (NWPU)-RESISC45 (NWPU) [45] dataset, University of California Merced (UCM) dataset [46], and aerial image data (AID) dataset [47]. These three datasets are divided into the training set, verification set, and test set as in [27]. The split results are shown in Table I.

The NWPU-RESISC45 dataset is a benchmark dataset for remote sensing scene classification constructed by Northwestern Polytechnical University, which contains 31 500 images with a size of 256 × 256 pixels. All images are organized in 45 scene classes, and each class has 700 images with RGB color space. The entire dataset is divided into 25, 10, and 10 classes for training, validation, and testing, respectively.

The UC Merced dataset contains 21 categories of scenes, each of which has 100 images with the size of 256 × 256. There are 2100 pictures in total. Furthermore, the UCM dataset is divided into ten training sets, five validation sets, and six test sets in our experiments.

The AID dataset was jointly proposed by Huazhong University of Science and Technology and Wuhan University in 2017, which contains 10 000 images of 30 scene classes, and each class has approximately 220–420 images. The pixel size of each image is 600 × 600 pixels. In the dataset, 16 classes are randomly chosen as the training set, seven classes as the validation set, and the remaining seven classes as the test set.

B. Hyperparameter Settings

All remote sensing scene images are resized to 128 × 128. To classify them, we randomly generate 200 episodes for the training process, 500 episodes for the validation process, and 2000 episodes for the testing process. In each episode, there are five categories of images for both support set and query set, each of which contains one or five samples, i.e., in the case of five-way one-shot, one sample is selected randomly for each class; in the case of five-way five-shot, each class contains five samples. In addition, there are 15 samples for each class in the query set. We use more samples (15 samples here) as the query set since there is no limitation on the number of testing samples in practice. Calculating losses multiple times using more samples in query set can make our model more robust and stable. Moreover, most of our comparison methods adopted this setting [23]–[28]. For the sake of fairness and convenience, we choose 15 samples per class as query set.

Our feature extractor contains six convolution layers, all of which use 3 × 3 convolution kernels. The number of the output channel of each convolution layer is $o_1 = 64$, $o_2 = 128$, $o_3 = 256$, $o_4 = 512$, $o_5 = 512$, and $o_6 = 256$; the number of nodes in the full connection layer of the channel attention module is the number of channels divided by 16. Through the test and trial procedure, we selected 5 × 5 and 7 × 7 convolution kernels for the spatial attention module (the experiment results are shown in Table II). In addition, we adopt 1 and 0.1 as the weighting coefficients of the few-shot classification loss and SCL loss, respectively.

TABLE II
COMPARISON OF DIFFERENT CONVOLUTION KERNEL COMBINATIONS

Kernel Combination	5-way Acc. (%)					
	NWPU		UCM		AID	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
(3, 5)	59.15±0.48%	77.35±0.76%	50.72±0.94%	67.46±0.54%	57.26±0.72%	73.12±0.84%
(3, 7)	57.55±0.92%	78.09±0.62%	51.19±0.63%	67.57±0.68%	58.49±0.61%	75.28±0.57%
(3, 5, 7)	59.77±1.06%	78.72±0.91%	50.58±0.77%	65.09±0.85%	55.27±0.69%	72.60±0.45%
(5, 7) - Ours	**62.21±1.12%**	**80.86±0.76%**	**51.37±0.79%**	**68.09±0.92%**	**59.46±0.96%**	**76.31±0.68%**

(3, 5) represents 3 × 3 and 5 × 5 convolution kernels used in our spatial attention module, (3, 7), (3, 5, 7), and (5, 7) are in the same way.

TABLE III
COMPARISON OF DIFFERENT NUMBERS OF SHOTS ON THE NWPU, UCM, AND AID DATASETS

Dataset	5-way Acc.		
	1-shot	5-shot	10-shot
NWPU	62.21±1.12%	80.86±0.76%	84.13±1.46%
UCM	51.37±0.79%	68.09±0.92%	74.67±1.63%
AID	59.46±0.96%	76.31±0.68%	78.12±0.95%

TABLE IV
FEW-SHOT CLASSIFICATION ACCURACY COMPARISON ON THE NWPU DATASET

Method	5-way Acc.	
	1-shot	5-shot
MatchingNet	38.23±0.75%	47.40±0.69%
ProtoNet	40.35±1.02%	69.55±0.55%
MAML	47.52±0.70%	62.58±0.43%
LLSR	51.43%	72.90%
Meta LSTM	47.53±0.80%	72.36±0.54%
Meta-SGD	60.58±0.94%	76.04±0.49%
RelationNet	61.95±1.73%	76.28±1.42%
DLA-MatchNet	67.85±0.68%	79.97±0.75%
RS-MetaNet	46.21±0.58%	68.75±0.70%
SCL-MLNet(ours)	62.21±1.12%	80.86±0.76%

In this article, we use Adam optimizer to optimize the feature extractor, the initial learning rate of which is 0.001 and the learning rate is updated every ten epochs. In addition, the multiplication factor γ is set to 0.2 for updating the learning rate.

C. Comparison With Other Few-Shot Classification Methods

For experimental comparison, we adopt several classical few-shot classification methods for remote sensing scenes, including metric-learning based methods such as MatchingNet [22], ProtoNet [23], and RelationNet [24]. MatchingNet and ProtoNet use predefined and fixed metrics, whereas RelationNet adopts a learnable non-linear comparator. In addition, gradient-descent-based methods such as MAML [26], Meta LSTM [27], and Mate-SGD [28] aim to train a model that is easy to fine-tune on the few-shot problem during the test phase. The above approaches all use four convolutional blocks as mentioned in original papers as embedding module. Each block consists of a 64-filter 3 × 3 convolution, a batch normalization layer, a ReLU nonlinearity, and a 2 × 2 max-pooling layer. We also conduct experiments on approaches that are original designed for remote sensing scene classification, including lifelong learning for scene recognition (LLSR) [29], DLA-MatchNet [2], and remote sensing-MetaNet [30]. DLA-MatchNet adopted five convolutional blocks, and we set the output channels according to [2]. Remote sensing-MetaNet used four convolutional blocks to learn feature representations. The input images of the above approaches are resized to 128 × 128 which are the same as our approach.

The measure metric used to evaluate the performance of our framework is defined as follows: in the test phase, each episode performs a C-way K-shot task, and the overall accuracy is computed by calculating the average accuracy of 2000 episodes. The classification performance of the model is measured by comparing the accuracy.

In most of our experiments, we take one and five shots as examples like most state-of-the-arts do [2], [22]–[29], since as the number of shots increase from one to five, the improvement of accuracies is more obvious than from five to ten. We show the comparison results of different numbers of shots on NWPU, UCM, and AID datasets in Table III. It can be observed from Table III that as the number of shots increases from five to ten, the performances are improved a lot. Especially for UCM dataset, its accuracy increases by more than 6%. From one to five, the accuracies improved dramatically for all datasets.

In real-world scenario, we can choose suitable number of shots according to the realistic need.

Tables IV–VI display the experimental results of the above nine few-shot learning methods and SCL-MLNet on the three datasets; all methods are trained based on meta-learning. Among them, LLSR adopts the experimental results in the original paper. Table IV shows the classification results for the NWPU dataset, and the results in the five-way five-shot scenario show that SCL-MLNet achieves the best accuracy of 80.86%, surpassing the second-best approach DLA-MatchNet by 0.89%.

Table V presents the accuracy results for the UCM dataset; according to the table, in the five-way one-shot scenario, the classification accuracy of DLA-MatchNet is higher than any other methods with a value of 52.76%, which is 1.39% higher than SCL-MLNet. However, our framework is superior to DLA-MatchNet by 3.52% in the case of five shots and the second-best method is ProtoNet in this scenario. As shown in

TABLE V
FEW-SHOT CLASSIFICATION ACCURACY COMPARISON ON THE UCM DATASET

Method	5-way Acc.	
	1-shot	5-shot
MatchingNet	34.28±0.39%	53.64±0.74%
ProtoNet	52.42±0.09%	67.93±1.01%
MAML	49.01±0.58%	61.44±0.63%
LLSR	39.47%	57.40%
Meta LSTM	47.43±1.22%	64.72±2.49%
Meta-SGD	50.21±2.31%	61.35±2.13%
RelationNet	48.74±1.33%	61.29±0.46%
DLA-MatchNet	52.76±0.83%	64.57±0.67%
RS-MetaNet	52.31±1.12%	67.21±0.44%
SCL-MLNet(ours)	51.37±0.79%	68.09±0.92%

TABLE VI
FEW-SHOT CLASSIFICATION ACCURACY COMPARISON ON THE AID DATASET

Method	5-way Acc.	
	1-shot	5-shot
MatchingNet	35.33±0.47%	56.41±0.68%
ProtoNet	54.79±0.60%	69.36±0.10%
MAML	47.98±0.72%	61.72±0.81%
LLSR	-	-
Meta LSTM	49.79±0.60%	67.25±0.89%
Meta-SGD	53.14±1.46%	66.94±1.20%
RelationNet	55.26±1.14%	72.81±0.93%
DLA-MatchNet	57.21±0.82%	73.45±0.61%
RS-MetaNet	53.37±0.56%	72.59±0.73%
SCL-MLNet(ours)	59.46±0.96%	76.31±0.68%

Table VI, for the AID dataset, given one sample of each class, the best performance is achieved by SCL-MLNet, surpassing the top 2 approach by 2.25%. In the five-shot scenario, our framework is 2.86% higher than the second-best method RelationNet. In addition, since no experiment on the AID dataset was conducted using LLSR in [29], our study did not include it.

In summary, of all six groups of experiments on the three datasets, SCL-MLNet achieves the best performance than any other few-shot classification methods, except the one-shot scenario on the UCM and NWPU dataset. By analyzing the experiment results, we can conclude that our approach is more suitable for classification task with slightly more samples (e.g., five samples per class). Moreover, our framework has fast convergence speed on UCM dataset, which can reach the accuracy of 99.7% on the training data after approximately ten epochs.

D. Effect of Embedding Network Architecture

To learn more detailed feature representations, we investigate the performance of different networks as the backbone of SCL-MLNet. The accuracy of each network is obtained by averaging 20 rounds of experiments.

Eleven models are used in our comparison experiments, which can be divided into four groups, i.e., the shallow network sets (four-conv-layer, six-conv-layer, and seven-conv-layer), VGG-based network sets (VGG11, VGG16, and VGG19), Resnet-based network sets (Resnet18, Resnet50, and Resnet152), and other networks (AlexNet, GoogleNet). To explore how the classification accuracy of datasets changes with the deepening of network depth, we conduct experiments on the VGG-based network sets and ResNet-based network sets. Most of the embedding models adopted by few-shot classification methods are shallow networks; thus, experiments on shallow network sets can not only investigate the effect of network depth on classification accuracy but also explore the number of layers with the highest classification accuracy in shallow models.

Figs. 5–7 show the results of different embedding networks used in our feature extractor on the three datasets of NWPU, UCM, and AID. From the three histograms, we can easily find that the six-conv-layer model has the highest accuracy on both NWPU and AID. However, for UCM, the accuracy of the four-conv-layer model is slightly higher than that of the six-conv-layer model, which is approximately 2% higher in the one-shot scenario and 5% higher in the five-shot scenario. On all three datasets, the performance of the seven-conv-layer model is poorer than that of the six-conv-layer in the

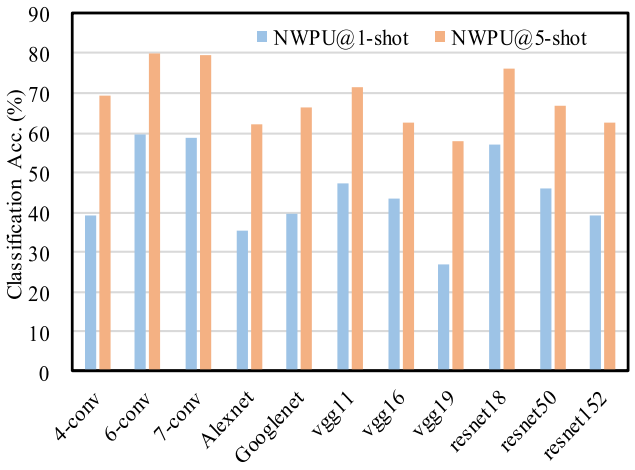

Fig. 5. Performance comparison of different embedding network architectures on the NWPU dataset.

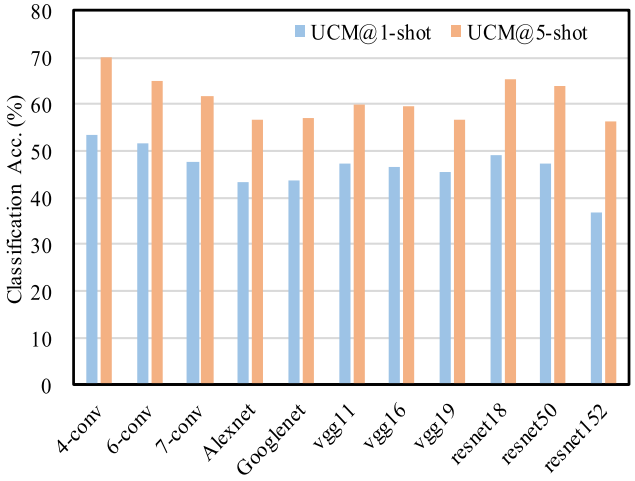

Fig. 6. Performance comparison of different embedding network architectures on the UCM dataset.

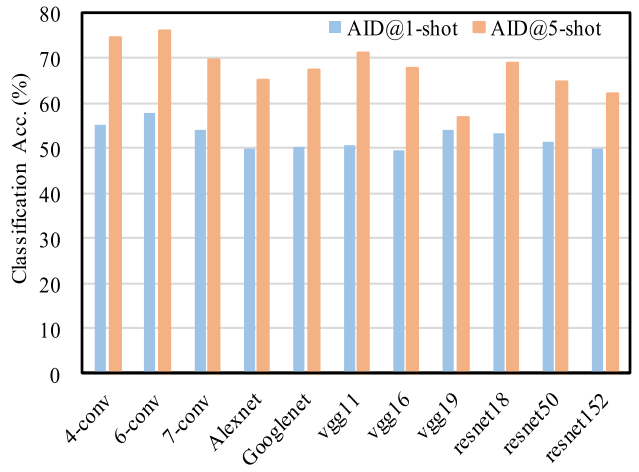

Fig. 7. Performance comparison of different embedding network architectures on the AID dataset.

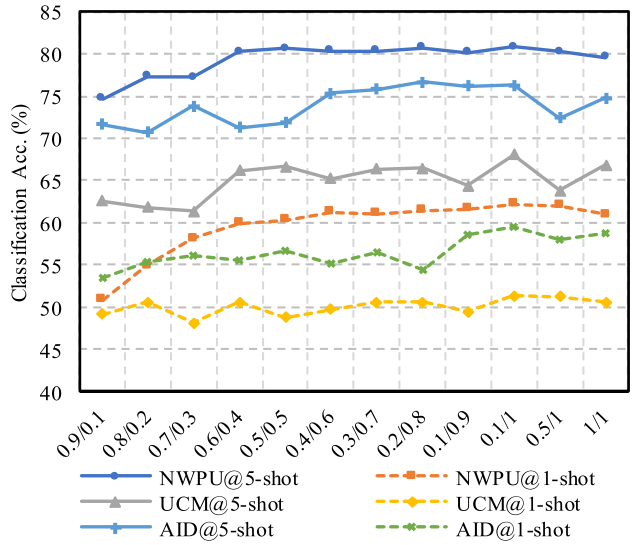

Fig. 8. Comparison of different weighting coefficients settings for self-supervised learning loss and few-shot classification loss on three datasets.

case of one and five shots. In addition, for the VGG-based network sets and Resnet-based network sets, as the number of network layers increases, the accuracy of these two types of models decreases to varying degrees, which is inferior to the six-conv-layer model by a large margin. Furthermore, it should be noted that when we use a VGG-based network as the embedding network, the results from multiple experiments greatly fluctuate, e.g., for the AID dataset, the accuracy of VGG16 fluctuates nearly 10% in the case of five-way one-shot. When we adopt VGG19 as the embedding network on the three datasets in the one-shot scenario, it gets stuck at a local optimal solution easily.

We can infer that using deep convolutional networks such as GoogleNet, VGGNet, and Resnet as the embedding network for remote sensing scene datasets cannot improve few-shot classification accuracy. Therefore, in our work, the six-conv-layer model with the attention module is chosen as the feature extractor of SCL-MLNet.

TABLE VII

CLASSIFICATION ACCURACY OF DIFFERENT ATTENTION MODULES ON THE NWPU DATASET

Method	5-way Acc.	
	1-shot	5-shot
SE-Conv	56.97%	80.05%
CBAM	61.88%	80.35%
SK-Conv	54.77%	73.55%
CSAM (ours)	62.21%	80.86%

TABLE VIII

CLASSIFICATION ACCURACY OF DIFFERENT ATTENTION MODULES ON THE UCM DATASET

Method	5-way Acc.	
	1-shot	5-shot
SE-Conv	49.89%	64.90%
CBAM	51.32%	65.72%
SK-Conv	49.83%	62.36%
CSAM (ours)	51.37%	68.09%

E. Selection of Weighting Coefficient

SCL-MLNet is trained under the joint supervision of SCL loss and few-shot classification loss. To obtain the optimal loss function, we conduct several experiments to choose appropriate weighting coefficients for the two parts of loss.

As shown in Fig. 8, the x-axis of the line chart represents the ratio of the SCL loss to the few-shot classification loss, namely, μ/λ, which are organized into 12 cases. We adopt two parameters λ and μ to balance these two parts of loss, because the sum of λ and μ is not a fixed value. The first nine cases ensure the sum of λ and μ is 1, and the rest of cases keep λ unchanged and then adjust μ. The solid lines are the five-shot classification accuracy of the three datasets, and the dashed lines are the one-shot classification accuracy. As shown in Fig. 8, λ rises from 0.1 to 0.4, and the broken lines also increase, that is, the accuracy increases in addition to the case of UCM@1-shot and AID@5-shot. When λ is between 0.6 and 0.9, except for the case of AID@1-shot, other broken lines remain stable, that is, the accuracy does not change much. Furthermore, all broken lines reach the peak value at $\mu = 0.1$, $\lambda = 1$.

From the results, we can conclude that of these two losses, the few-shot classification loss needs to account for a greater proportion. Therefore, we choose 0.1 as the weighting coefficient of the SCL loss and 1 is used as the weighting coefficient of the few-shot classification loss.

F. Comparison of Attention Module

To demonstrate the effectiveness of our attention module for feature learning, we compare CSAM with some common attention modules, e.g., SE-Conv [36], convolutional block attention module (CBAM) [48], and selective kernel convolution (SK-Conv) [49].

Tables VII–IX show the classification results of weaving above attention modules into the backbone of SCL-MLNET on three remote sensing scene datasets. According to the results,

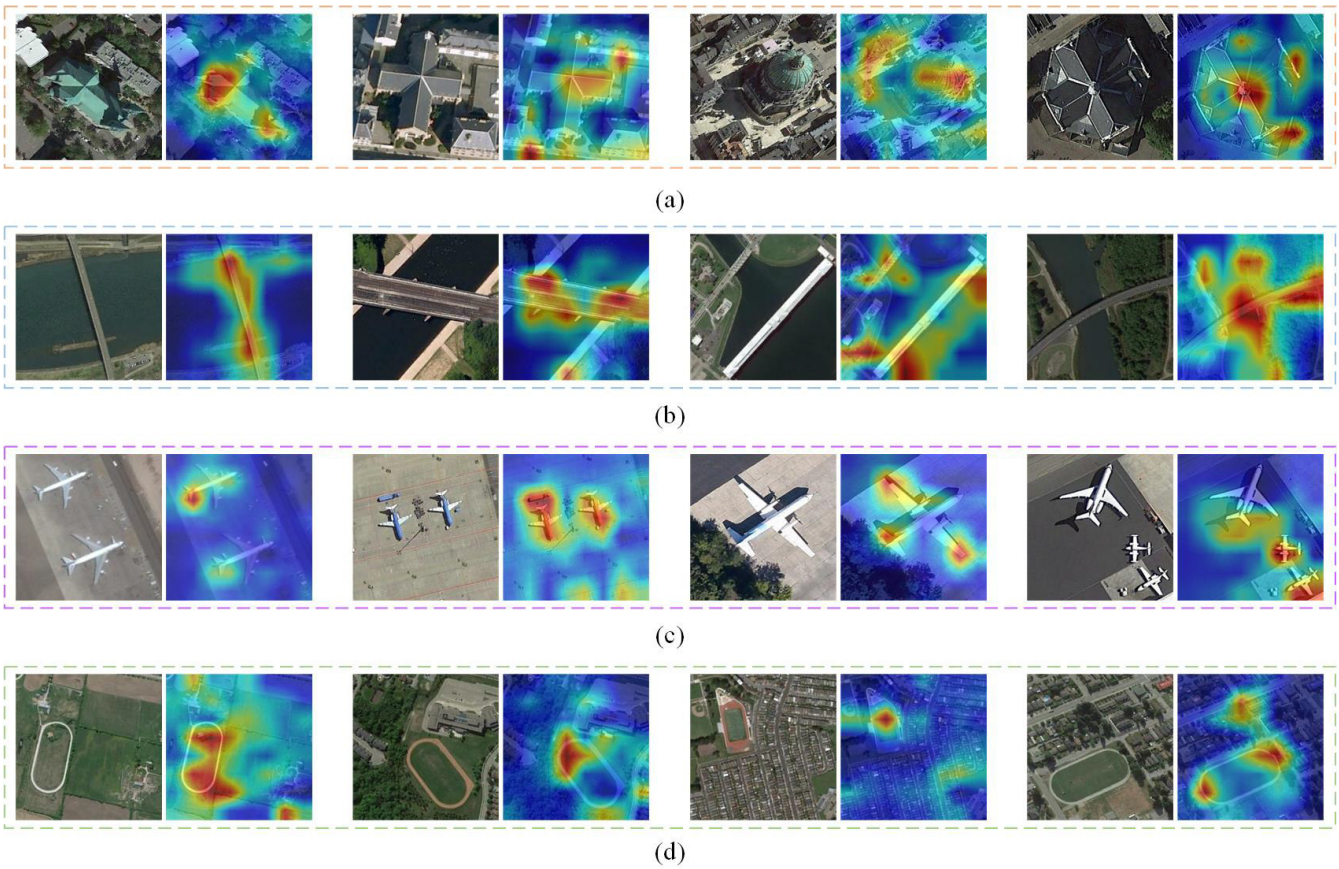

Fig. 9. Comparison of the original image and class-discriminative representation visualization heat map. (a) Church. (b) Bridge. (c) Airplane. (d) Ground track field.

TABLE IX
CLASSIFICATION ACCURACY OF DIFFERENT ATTENTION MODULES ON THE AID DATASET

Method	5-way Acc.	
	1-shot	5-shot
SE-Conv	58.06%	76.29%
CBAM	56.58%	72.01%
SK-Conv	54.77%	73.67%
CSAM (ours)	59.46%	76.31%

in the one- and five-shot scenarios, the attention module with the highest accuracy on both NWPU and UCM datasets is CSAM, followed by CBAM, and SK-Conv has the worst accuracy. In addition, on the AID dataset, CSAM achieves the best performance, followed by SE-Conv.

Overall, regardless of the one-shot scenario or the five-shot scenario, our attention module CSAM always achieves better performance than any other attention modules. Moreover, SK-Conv achieves the worst performance in all cases, except on the AID dataset in the case of five-way five-shot.

Therefore, based on the above results, our CSAM attention module along with the six-conv-layer backbone can learn more detailed feature representations, which is appropriate for remote sensing scene classification tasks.

Furthermore, to show the effectiveness of the feature extractor in SCL-MLNet more intuitively, we illustrate the feature representation heat map according to Grad-class activation mapping (CAM) [50] to visualize the class-discriminative representation and make our embedding network more transparent and explainable [34].

As shown in Fig. 9, SCL-MLNet can exactly capture the class-discriminative features, and the red area of the figure is the exact location of the object having the most important feature. The darker the red color, the more the area contributes to the final classification, and the darker the blue color, the lesser the contribution to the final classification.

The heat map of Fig. 9(c) shows that for images whose areas were largely occupied by airplanes, our embedding model focused on the tail and wings of the airplanes. Similarly, for images whose areas are scarcely occupied by airplanes, the model can accurately concentrate on the entire airplanes. In addition, according to the second pair of images of the airplane and the third pair of images of the ground track field in Fig. 9, it can be concluded that the Grad-CAM visualization method can be used to solve the small target detection problem.

G. Ablation Study

In this section, we conduct ablation studies to explore the roles of the CSAM attention module and SCL task. As shown in Tables X–XII, *Backbone* denotes a six-layer convolutional network, abbreviated as B; B+SCL means that we use a six-layer convolution network as the embedding

TABLE X
ABLATION STUDY OF SCL-MLNET ON THE NWPU DATASET

Method	5-way Acc. (%)	
	1-shot	5-shot
Backbone	58.5	79.84
B + SCL	59.4	79.91
B + CSAM	58.88	77.72
B + CSAM + SCL	**62.21**	**80.86**

TABLE XI
ABLATION STUDY OF SCL-MLNET ON THE UCM DATASET

Method	5-way Acc. (%)	
	1-shot	5-shot
Backbone	50.18	64.93
B + SCL	51.45	65.01
B + CSAM	**51.70**	67.72
B + CSAM + SCL	51.37	**68.09**

TABLE XII
ABLATION STUDY OF SCL-MLNET ON THE AID DATASET

Method	5-way Acc. (%)	
	1-shot	5-shot
Backbone	57.62	70.05
B + SCL	57.84	76.15
B + CSAM	55.75	69.17
B + CSAM + SCL	**59.46**	**76.31**

network, and we weave SCL into the first stage of few-shot learning. *B+CSAM+SCL* represents SCL-MLNet proposed in this article, which adds an extra attention module to the *Backbone* based on *B+SCL*.

In all the cases, the accuracy of *B+SCL* is higher than Backbone, especially in the case of five-way five-shot on AID dataset, its accuracy is 6.1% higher than Backbone. For NWPU and AID datasets, the classification accuracy of the model increases with the addition of SCL and CSAM (*B+SCL+CSAM*), whether in the case of one or five shots, and the classification accuracy of *B+SCL* is higher than *B+CSAM* in all the cases, especially in the case of five shots. For the UCM dataset, *B+CSAM* is slightly higher than *B+SCL* in the one-shot scenario and surpasses *B+SCL* by 2.71% in the five-shot scenario. For all the three datasets, once we add *SCL* into *B+CSAM* (that is *B+CSAM+SCL*), its accuracy is superior to *B+SCL* and *B+CSAM* in all scenarios except in the case of one shot for the UCM dataset.

In conclusion, it is effective to introduce SCL into few-shot classification and add the attention mechanism into the feature extractor.

V. CONCLUSION

This article proposes an end-to-end few-shot learning framework for remote sensing scene classification called SCL-MLNet. Our method weaves SCL task into few-shot learning approaches, which can fully use the limited annotated data in the real world. We introduce a new loss function to balance the SCL loss and few-shot classification loss. SCL-MLNet can also fuse multi-scale features extracted by our novel attention module. We conduct comparison experiments on three publicly remote sensing scene classification datasets, and the experiment results show that our method can obtain representative features and outperforms other state-of-the-art few-shot classification methods.

Nonetheless, another challenge exists, which is the unavailability of unannotated remote sensing scene data for common few-shot classification methods. Therefore, in our future work, we will fully use unannotated data in the SCL tasks, adopting both annotated and unannotated data as the input of SCL-MLNet to further improve the model's performance and show the value of unannotated data.

REFERENCES

[1] H. Sun, S. Li, X. Zheng, and X. Lu, "Remote sensing scene classification by gated bidirectional network," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 1, pp. 82–96, Jan. 2020.

[2] L. Li, J. Han, X. Yao, G. Cheng, and L. Guo, "DLA-MatchNet for few-shot remote sensing image scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 9, pp. 1–10, Sep. 2020.

[3] J. Zhang, J. Liu, B. Pan, and Z. Shi, "Domain adaptation based on correlation subspace dynamic distribution alignment for remote sensing image scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 11, pp. 7920–7930, Nov. 2020.

[4] Y. Li, H. Zhang, X. Xue, Y. Jiang, and Q. Shen, "Deep learning for remote sensing image classification: A survey," *Wiley Interdiscip. Rev., Data Min. Knowl. Discov.*, vol. 8, no. 6, p. e1264, 2018.

[5] G. Cheng, X. Xie, J. Han, L. Guo, and G.-S. Xia, "Remote sensing image scene classification meets deep learning: Challenges, methods, benchmarks, and opportunities," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, no. 99, pp. 3735–3756, Jun. 2020.

[6] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state of the Art," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.

[7] X. Yu, X. Wu, C. Luo, and P. Ren, "Deep learning in remote sensing scene classification: A data augmentation enhanced convolutional neural network framework," *GISci. Remote Sens.*, vol. 54, no. 5, pp. 741–758, Sep. 2017.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.

[10] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

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

[12] S. Xie, R. Girshick, P. Dollar, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1492–1500.

[13] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele, "Meta-transfer learning for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 403–412.

[14] S. Qiao, C. Liu, W. Shen, and A. Yuille, "Few-shot image recognition by predicting parameters from activations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7229–7238.

[15] W. Chen, Y. Liu, Z. Kira, Y. Wang, and J. Huang, "A closer look at few-shot classification," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–17.

[16] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–34, 2020.

[17] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.

[18] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1842–1850.

[19] M. Ren *et al.*, "Meta-learning for semi-supervised few-shot classification," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018.

[20] Y. Bengio, S. Bengio, and J. Cloutier, "Learning a synaptic learning rule," in *Proc. Seattle Int. Joint Conf. Neural Netw. (IJCNN)*, 1991, p. 969.

[21] L. Fe-Fei, R. Fergus, and P. Perona, "A Bayesian approach to unsupervised one-shot learning of object categories," in *Proc. 9th IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 1134–1141.

[22] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.

[23] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.

[24] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.

[25] X. Zhang, Y. Qiang, F. Sung, Y. Yang, and T. Hospedales, "RelationNet2: Deep comparison network for few-shot learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[26] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1126–1135.

[27] A. Srinivasan, A. Bharadwaj, M. Sathyan, and S. Natarajan, "Optimization of image embeddings for few shot learning," in *Proc. 10th Int. Conf. Pattern Recognit. Appl. Methods*, 2017, pp. 236–242.

[28] Z. Li, F. Zhou, F. Chen, and H. Li, "Meta-SGD: Learning to learn quickly for few-shot learning," 2017, *arXiv:1707.09835*. [Online]. Available: https://arxiv.org/abs/1707.09835

[29] M. Zhai, H. Liu, and F. Sun, "Lifelong learning for scene recognition in remote sensing images," *IEEE Geosci. Remote Sens. Lett.*, vol. 16, no. 9, pp. 1472–1476, Sep. 2019.

[30] H. Li *et al.*, "RS-MetaNet: Deep meta metric learning for few-shot remote sensing scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 8, pp. 1–12, Aug. 2020.

[31] M. Noroozi, A. Vinjimoor, P. Favaro, and H. Pirsiavash, "Boosting self-supervised learning via knowledge transfer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9359–9367.

[32] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, May 4, 2020, doi: 10.1109/TPAMI.2020.2992393.

[33] S. Gidaris, A. Bursuc, N. Komodakis, P. P. Perez, and M. Cord, "Boosting few-shot visual learning with self-supervision," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8059–8068.

[34] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, Dec. 2020.

[35] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, "Scaling and benchmarking self-supervised visual representation learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6391–6400.

[36] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze- and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2017, pp. 7132–7141.

[37] G. Cheng, C. Yang, X. Yao, L. Guo, and J. Han, "When deep learning meets metric learning: Remote sensing image scene classification via learning discriminative CNNs," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 5, pp. 2811–2821, May 2018.

[38] J. Xie, N. He, L. Fang, and A. Plaza, "Scale-free convolutional neural network for remote sensing scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6916–6928, Sep. 2019.

[39] X. Lu, H. Sun, and X. Zheng, "A feature aggregation convolutional neural network for remote sensing scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 10, pp. 7894–7906, Oct. 2019.

[40] K. Xu, H. Huang, P. Deng, and G. Shi, "Two-stream feature aggregation deep neural network for scene classification of remote sensing images," *Inf. Sci.*, vol. 539, pp. 250–268, Oct. 2020.

[41] Y. Xu, B. Du, and L. Zhang, "Assessing the threat of adversarial examples on deep neural networks for remote sensing scene classification: Attacks and defenses," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 2, pp. 1604–1617, Feb. 2021.

[42] J.-B. Grill *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," in *Proc. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 21271–21284.

[43] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.

[44] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. Workshop Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–4.

[45] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: Benchmark and state of the art," *Proc. IEEE*, vol. 105, no. 10, pp. 1865–1883, Oct. 2017.

[46] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*, 2010, pp. 270–279.

[47] G.-S. Xia *et al.*, "AID: A benchmark data set for performance evaluation of aerial scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 7, pp. 3965–3981, Jul. 2017.

[48] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 3–19.

[49] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 510–519.

[50] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

Xiaomin Li received the B.E. degree in software engineering from Jilin University, Jilin, Changchun, China, in 2019, where she is currently pursuing the M.S. degree with the College of Computer Science and Technology.

Her research interests include computer vision, self-supervised learning, and few-shot learning.

Daqian Shi received the B.Sc. degree from Shanghai University, Shanghai, China, in 2017, and the M.Res. degree from the University of Manchester, Manchester, U.K., in 2018. He is currently pursuing the Ph.D. degree with the Department of Information Engineering and Computer Science, University of Trento, Trento, Italy.

His research interests include knowledge representation, computer vision, and artificial intelligence.

Xiaolei Diao received the B.Sc. and M.Sc. degrees from the College of Computer Engineering and Science, Shanghai University, Shanghai, China, in 2017 and 2020, respectively. She is currently pursuing the Ph.D. degree with the Department of Information Engineering and Computer Science, University of Trento, Trento, Italy.

Her research interests include machine learning and computer vision.

Hao Xu received the B.S. and M.S. degrees in computer science from Jilin University, Jilin, Changchun, China, in 2005 and 2008, respectively, and the Ph.D. degree from the University of Trento, Trento, Italy, in 2011.

He is a Professor with Jilin University. His research interests include human-centered artificial intelligence, knowledge representation, and computer vision.