Generalized Scene Classification From Small-Scale Datasets With Multitask Learning

Xiangtao Zheng, *Member, IEEE*, Tengfei Gong, Xiaobin Li, and Xiaoqiang Lu, *Senior Member, IEEE*

***Abstract*— Remote sensing images contain a wealth of spatial information. Efficient scene classification is a necessary precedent step for further application. Despite the great practical value, the mainstream methods using deep convolutional neural networks (CNNs) are generally pretrained on other large datasets (such as ImageNet) and thus fail to capture the specific visual characteristics of remote sensing images. For another, it lacks the generalization ability to new tasks when training a new CNN from scratch with an existing remote sensing dataset. This article addresses the dilemma and uses multiple small-scale datasets to learn a generalized model for efficient scene classification. Since the existing datasets are heterogeneous and cannot be directly combined to train a network, a multitask learning network (MTLN) is developed. The MTLN treats each small-scale dataset as an individual task and uses complementary information contained in multiple tasks to improve generalization. Concretely, the MTLN consists of a shared branch for all tasks and multiple task-specific branches with each for one task. The shared branch extracts shared features for all tasks to achieve information sharing among tasks. The task-specific branch distills the shared features into task-specific features toward the optimal estimation of each specific task. By jointly learning shared features and task-specific features, the MTLN maintains both generalization and discrimination abilities. Two types of MTL scenarios are explored to validate the effectiveness of the proposed method: one is to complete multiple scene classification tasks and the other is to jointly perform scene classification and semantic segmentation.**

***Index Terms*— Attention, multitask learning (MTL), scene classification, small-scale dataset.**

I. Introduction

THE development of remote sensing technology makes it easier to obtain high spatial resolution remote sensing images [1], [2]. These images contain rich spatial information, which is very helpful for geological surveys [3], urban planning [4], and other fields [5], [6]. As a necessary precedent step, remote sensing scene classification assigns a specific semantic label to each image [7].

Currently, the deep convolutional neural networks (CNNs) are the mainstream methods for remote sensing scene classification [8], [9]. Early works train a CNN from scratch with the remote sensing scene data of interest [10]. Unfortunately, training CNN needs a large amount of labeled data, while manually annotating remote sensing scenes is time-consuming and laborious [11]. A simple solution is to pre-train CNN on large-scale datasets (such as ImageNet [12]). Then, the pretrained CNN can be directly used as a feature extractor by removing the classification layer [2]. Furthermore, the pretrained CNN can be fine-tuned on target remote sensing datasets [13]–[15]. However, directly using the pretrained CNN fails to capture the specific characteristics of the remote sensing image: the imaging perspective and object distribution [16]. Fine-tuning may easily stick in local minimum with a small-scale remote sensing dataset because of the millions of parameters contained in CNNs.

This article is motivated by the study of Torralba and Efros [17], which concerns the generalization ability across different datasets and claims that even the same task has different distributions of features in different datasets. To this end, we consider how to learn a generalized CNN by exploiting multiple small-scale remote sensing datasets. With the learned generic knowledge, the model can quickly adapt to new unseen tasks without numerous labeled samples. There are many applications especially for the case of implementing a new task with limited training data. For example, the model can be explored to implement classification among different domains, such as the agricultural classification [3], forestry classification [18], and urban classification [4]. In addition, with the increasing satellite data every day, instantly classifying the new data by exploiting historical data is also a valuable application [19].

The existing remote sensing datasets such as Merced [20], AID [1], and NWPU [21] allow to train a model for an executed task, while lack the generalization ability to other tasks. Combining these multiple related small-scale datasets exhaustively may be enough to train a generalized network as the pretrained network. However, it is difficult to train a network with these multiple datasets directly, since each small-scale dataset is annotated for a different task. These datasets are heterogeneous in terms of different data distributions, distinct label spaces, various image sizes, etc. [22].

Manuscript received February 26, 2021; revised July 9, 2021 and September 2, 2021; accepted September 19, 2021. This work was supported in part by the National Science Fund for Distinguished Young Scholars under Grant 61925112, in part by the National Natural Science Foundation of China under Grant 61806193 and Grant 61772510, in part by the Innovation Capability Support Program of Shaanxi under Grant 2020KJXX-091 and Grant 2020TD-015, in part by the Natural Science Basic Research Program of Shaanxi under Grant 2019JC-23, and in part by the Chinese Association for Artificial Intelligence (CAAI)-Huawei MindSpore Open Fund. *(Corresponding author: Xiaoqiang Lu.)*

Xiangtao Zheng and Xiaoqiang Lu are with the Key Laboratory of Spectral Imaging Technology CAS, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, Shaanxi 710119, China (e-mail: luxq666666@gmail.com).

Tengfei Gong is with the Key Laboratory of Spectral Imaging Technology CAS, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, Shaanxi 710119, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China.

Xiaobin Li is with Beijing Institute of Remote Sensing Information, Beijing 100192, China.

Digital Object Identifier 10.1109/TGRS.2021.3116147

To circumvent the dilemma, an efficient multitask learning network (MTLN) is designed in this article. The MTLN treats each small-scale dataset as a task and designs multiple task-specific branches to learn multiple tasks simultaneously. Each task corresponds to one task-specific branch, but all tasks explore a shared branch for complementary information sharing. By jointly optimizing the shared branch and task-specific branches, the MTLN maintains both generalization and discrimination abilities. In this article, two types of multitask learning (MTL) scenarios are explored: multiple-scene classification (MSC) and pixel-scene classification (PSC). MSC considers multiple remote sensing scene classification by exploiting multiple small-scale datasets. PSC jointly performs pixel-level semantic segmentation and image-level scene classification.

The MTLN contains three main steps: shared feature extraction, task-specific feature distillation, and multitask classification. First, the shared branch exploits convolution blocks to generate shared features for all tasks, which allows complementary information shared among tasks. Second, each task-specific branch is assigned for one task for task-specific feature distillation. The task-specific branch distills shared features into task-specific features through several cascaded attention modules. Each attention module generates an attention mask to strength features important to the executed task while ignoring the unimportant ones. Multiple attention modules are used to attend features adaptively as layers go deeper. Finally, after a sequence of attention modules, the output of the last attention module is passed to a task-specific classifier. Experiments are conducted on two MTL scenarios. To validate the PSC scenario, two PSC datasets are constructed by improving an existing dataset, with each image containing pixel-level semantic segmentation labels and an image-level scene label.

In summary, the main contributions of this article are listed in the following.

1) This article exploits multiple small-scale remote sensing datasets to learn a generalized CNN, which alleviates the difficulty of training CNNs with limited labeled samples.
2) An MTLN is proposed to incorporate complementary information among tasks and ensure an optimal result for each individual task.
3) To validate the proposed method, two types of MTL scenarios are explored: MSC and PSC. Two PSC datasets are built to identify the scene with semantic segmentation. The datasets are available at https://github.com/spectralpublic/GID-MTL.

The rest of this article is organized as follows. Section II reviews the related work. Section III discusses the proposed method in detail. Section IV provides the experiments. Section V concludes this article.

II. Related Work

A. Remote Sensing Scene Classification

Remote sensing scene classification assigns a specific scene label to an image and is a necessary precedent step for further application. Currently, remote sensing scene classification has achieved high accuracy, thanks to the deep CNNs. The strategies of exploiting CNNs can generally be grouped into three types: 1) training new CNNs from scratch; 2) using pretrained CNNs as feature extractors; and 3) fine-tuning pretrained CNNs.

The first strategy fully trains a new CNN from scratch with remote sensing images. Since a large-scale CNN contains millions of parameters, millions of training data are needed to fully train the network [12], [23]. However, in the remote sensing field, usually only hundreds or thousands of labeled images are available [21]. Training CNNs from scratch is not widely used in remote sensing.

The second strategy which uses pretrained CNNs as feature extractors has gained increasing popularity. It benefits from the characteristic that the initial layers of CNNs tend to capture generic structures, like edge or color, which are shared for all visual tasks. The widely used pretrained CNNs contain AlexNet [24], VGGNet [25], and ResNet [23]. Penatti *et al.* [26] take outputs of the last convolutional layer of a pretrained CNN as global features. Zheng *et al.* [2] further perform a multiscale strategy over the CNN activations. However, the pretrained CNN may fail to obtain the specific feature of remote sensing images due to the semantic gap between natural images and remote sensing images [16].

The third strategy uses the pretrained CNN for classification by fine-tuning the target remote sensing dataset [13], [15]. Usually, the parameters of the earlier layers are fixed, and the final layers are retrained to obtain specific features of the target data. Nogueira *et al.* [16] explore remote sensing scene classification performances on multiple CNNs and point that fine-tuning achieves the best performance compared with full training and using pretrained CNNs as feature extractors. Based on that, some recent studies are developed to further improve the performance. For example, Othman *et al.* [27] exploit domain adaptation techniques to reduce the distribution shift between pretrained and target datasets. Although effective, the fine-tuning strategy still relies on the pretrained model and cannot adapt to specific or complex scenarios [26].

B. MTL

MTL improves generalization performance by simultaneously conducting multiple related tasks and exploring shared knowledge among them [28]–[30]. The performance of one task can be improved using related tasks as inductive bias [31], [32]. For example, Liu *et al.* [5] verify that scene classification and aerial image retrieval tasks can be regularized mutually and promote each other when learned simultaneously. Qiu *et al.* [33] combine classification and regression tasks jointly and exploit the prediction of the regression task as a prior for the classification task. Guo *et al.* [34] achieve effective building extraction with the regularization of the scene classification task. Taskonomy explores 26 tasks to reuse the supervision among related tasks and further reduce the need for labeled data [35]. Zhang *et al.* [7] explore the latent visual interactions among multiple related image categories to promote feature learning on aerial image classification tasks.

Most existing MTL methods can be roughly divided into two groups: hard-parameter sharing and soft-parameter sharing. Hard-parameter sharing exploits the same initial

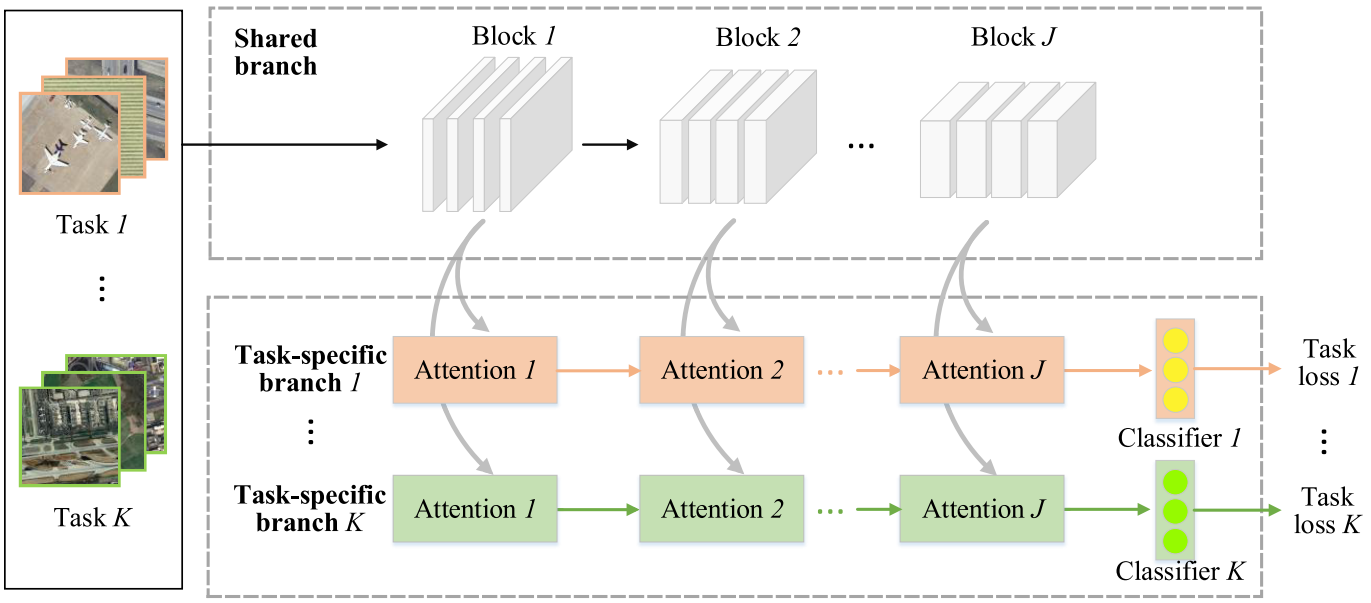

Fig. 1. Framework of the proposed method. The shared branch extracts shared features for all tasks. Task-specific branch distills needed information with attention modules to finish the executed task. Each task-specific branch contains multiple attention modules, and each module latches onto one block of the shared branch.

layers and different specialized classifier branches for different tasks [36]–[38]. However, due to different tasks concerning different information (such as the high-level semantic information for scene classification and low-level spatial information for segmentation), using the exact same features for all tasks may cause task interference [39]. Soft-parameter sharing allows each task to have task-adaptive model and the information can be shared among different tasks. For example, Zhang *et al.* [40] design two different networks for different tasks (a multilayer perception for land-cover classification and an object-based CNN for land-use classification) and refine one task with a prediction from another task through iteration to allow information sharing. Maninis *et al.* [42] use a unified network to perform multiple tasks, but exploit the attention module [41] to enable the network to adaptively learn different features according to the task currently being performed. Sun *et al.* [43] select specific layers for an executed task in a multitask network, in which a task-specific select-or-skip policy is explored. In this article, a shared branch and multiple task-specific branches are used to achieve information sharing among tasks and avoid interference between tasks.

III. Method

This article aims to learn generalized features that can be well-adapted to different scene classification tasks. Given K tasks with training data $\{X_k, Y_k\}_{k=1}^{K}$, where X_k and Y_k represent the N_k training images and corresponding labels, respectively, of the kth task, the MTLN takes full use of the complementary information from K tasks and distills effective features for multitask classifications. Concretely, the MTLN is conducted in three steps: first, a shared branch receives images from all tasks and provides shared features for all tasks. Second, each task-specific branch distills the shared feature into task-specific features for the executed task. Finally, the task-specific feature is passed to the task-specific classifier for the final classification. A clear flowchart is shown in Fig. 1.

A. Shared Feature Extraction

All tasks exploit a shared branch to extract shared features, which allows complementary visual features to be shared among tasks. The shared branch is a deep CNN that is stacked by J convolution blocks. Many classical network structures can be used, such as VGG or ResNet. The shared branch provides a field of intermediate layer features instead of a single classification result at the end of the network. We refer to the learned shared features as $\{\mathbf{f}_k^j\}_{j=1}^{J}$

$$\{\mathbf{f}_k^j\}_{j=1}^{J} = F(x_k) \tag{1}$$

where $F(\cdot)$ denotes the feature extraction with the shared branch. $x_k \in X_k$ denotes an image from the kth dataset X_k. $\mathbf{f}_k^j$ denotes the extracted jth block feature from the image x_k.

B. Task-Specific Feature Distillation

After shared features are learned, each task-specific branch distills the shared features for a specific task. The task-specific branch contains a set of attention modules, and each attention module latches onto one block of the shared branch. The attention module generates an attention mask that softly weights the shared feature to learn a task-specific feature, as shown in Fig. 2. Multiple attention modules are used to capture different attention information from the shared branch. Information is collected sequentially and used to determine what to attend in the next task-specific feature learning steps.

Given the shared branch output $\mathbf{f}_k^j$ from the jth block, the task-specific feature in this block is obtained by applying

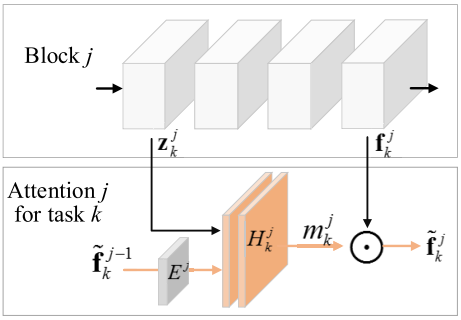

Fig. 2. Attention module learns the task-specific feature by weighting the shared feature with a generated attention mask.

an attention mask to the shared feature

$$\tilde{\mathbf{f}}_k^j = \mathbf{f}_k^j \odot m_k^j \tag{2}$$

where $\tilde{\mathbf{f}}_k^j$ is the learned task-specific feature in block j for task k. m_k^j is the corresponding attention mask. $\odot$ identifies elementwise multiplication. The attention module acts as a selector to determine the shared feature of the current block being used or not.

The attention mask m_k^j is generated by an attention module, which uses convolutional operation to transfer input features into an attention mask. The first attention module takes as input the feature from the first block of the shared branch. The subsequent attention module takes as input the concatenation of the feature from the shared branch and task-specific feature obtained from the previous block. The attention mask is formulated as

$$m_k^j = \begin{cases} H_k^j(\mathbf{z}_k^j), & j = 1 \\ H_k^j([\mathbf{z}_k^j; E^j(\tilde{\mathbf{f}}_k^{j-1})]), & j > 1 \end{cases} \tag{3}$$

where $\mathbf{z}_k^j$ is a feature from a different layer of the same feature blocks as $\mathbf{f}_k^j$ in the shared branch. H_k^j is the attention module in the jth block for the kth task, which is composed of a stack of convolution layers. E^j denotes an extractor for passing task-specific features from the previous block to the current jth block. The extractor is a convolutional layer in which downsampling is used to keep the size consistent. Different extractors $\{E^j\}_{j=2}^J$ are assigned to different blocks, but the same extractor E^j is shared for all tasks, which avoids parameter increasing linearly with the number of tasks. Task-specific features are learned by attention modules instead of extractors.

The attention module can make the feature $\mathbf{f}_k^j$ totally used for the kth task by assigning higher values to m_k^j, or only choosing the needed task-specific feature by assigning lower values. In this way, learning task-specific features from the shared features while sharing information across different tasks can be achieved simultaneously.

C. Multitask Classification and Optimization

After a sequence of attention modules, the output of the last attention module $\tilde{\mathbf{f}}_k^J$ is passed to the task-specific classifier for final prediction

$$\hat{y}_k = G_k(\tilde{\mathbf{f}}_k^J) \tag{4}$$

where $\hat{y}_k$ is the prediction of image x_k, and G_k is the specific classifier for the kth task. The task-specific classifier is designed based on the specific task. The MTLN is optimized by minimizing losses over multiple tasks. A simple linear sum is used to combine the K task-specific losses L_k

$$\min_{F,E,\{H_k,G_k\}_{k=1}^K} \mathcal{L} = \sum_{k=1}^K L_k \tag{5}$$

where $E = \{E^j\}_{j=2}^J$ and $H_k = \{H_k^j\}_{j=1}^J$ contain extractors and attention modules for J blocks, respectively. The parameters of the shared branch F, extractors E, attention modules H_k, and task-specific classifiers G_k in K branches are jointly optimized with the losses L_k from K tasks. Therefore, features in the shared branch can be shared to the greatest extent, while maximized task-specific performances are encouraged with the task-specific branches.

This work explores two MTL scenarios of incorporating related information. The first scenario implements MSC tasks. The second scenario simultaneously implements scene classification and semantic segmentation. For the scene classification task, each image is assigned to a scene category. A single fully connected layer following an average pooling layer is used as a scene classifier. For the semantic segmentation task, each pixel in an image is assigned to a category. A pixel classifier is constructed, which is symmetric to the shared branch. The shared branch and the pixel classifier form an encoder–decoder pattern as SegNet [44]. For the scene classification task, the cross-entropy loss is used

$$L_k = -\frac{1}{N_k} \sum_{(x_k,y_k)\in(X_k,Y_k)} y_k \log \hat{y}_k \tag{6}$$

where y_k is the label of image x_k. For semantic segmentation which is essentially a pixel-level classification task, pixelwise cross-entropy is used

$$L_k = -\frac{1}{N_k} \sum_{(x_k,y_k)\in(X_k,Y_k)} \frac{1}{N} \sum_{n=1}^N y_k^n \log \hat{y}_k^n \tag{7}$$

where y_k^n and $\hat{y}_k^n$ denote the semantic segmentation label and prediction of image x at pixel n, respectively. N is the total number of pixels.

IV. Experiments

This section presents the datasets, experimental results, along with the analysis of the proposed method on the MTL scenarios.

A. Experimental Setup

1) MTL Scenarios: We explore two MTL scenarios: MSC and PSC. The MSC exploits MSC datasets and each dataset is considered as a task. The PSC predicts two different formats of label (image-level scene category and pixel-level semantic segmentation map) simultaneously for each image. It is believed that scene classification and semantic segmentation can promote each other when implemented simultaneously.

TABLE I
SUMMARY OF THE DATASETS. #IMAGES AND #CLASSES DENOTE THE NUMBER OF IMAGES AND CLASSES IN A DATASET

Dataset	Source	Resolution	Size	#Images	#Classes
Merced [20]	US Geological Survey National Map	0.3m	256×256	2,100	21
AID [1]	Google Earth imagery	0.5∼8m	600×600	10,000	30
NWPU [21]	Google Earth imagery	0.2∼30m	256×256	31,500	45
PatternNet [45]	Google Earth imagery	0.1∼4.7m	256×256	30,400	38
EuroSAT[46]	Sentinel-2 satellite imagery	10m	64×64	27,000	10
GID [3]	Gaofen-2 satellite imagery	4m	6800×7200	160	5&15

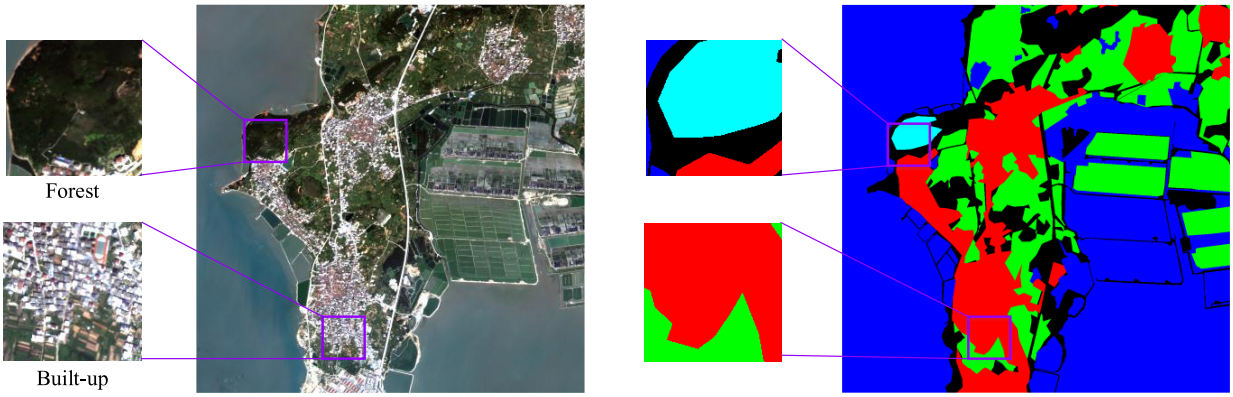

Fig. 3. Way of constructing PSC datasets PSC5 and PSC15. (a) Cropping sub-images from large-scale images of GID and annotating a scene label for each sub-image. (b) Obtaining pixel-level annotations by cropping from the original annotation.

2) Datasets: Six datasets divided into three groups are exploited to evaluate the proposed method. First, three remote sensing scene classification datasets-Merced,[1] AID,[2] and NWPU[3]-are used to evaluate MSC. Second, GID[4] which is a pixel-level labeled land-cover classification dataset is expanded to evaluate PSC. Third, to validate the generalization ability of the proposed method, the pretrained MTLNs are tested on new datasets. Two additional remote sensing scene classification datasets-PatternNet[5] and EuroSAT[6]-are exploited. The detailed information about these datasets is listed in Table I.

For the MSC, Merced, AID, and NWPU are selected. These datasets are composed of RGB images, but vary in complexity and difficulty. Different datasets have various resolutions and are associated with different scene categories, and therefore are viewed as different tasks. These diversities aim to illustrate the generalization ability of the proposed model. These datasets are organized to form an MSC dataset and a mini MSC (min-MSC) dataset. These two different datasets are designed to validate the performance of the proposed method under different levels of complexity. The MSC dataset contains all images from Merced, AID, and NWPU. The mini-MSC dataset contains only 12 common classes shared by Merced, AID, and NWPU. Moreover, each class contains 100 images for each task. The mini-MSC dataset has less interference among tasks because of eliminating the impact of different classes and a variable number of images per class. The selected 12 common classes are *airfield, anchorage, beach, dense residential, farm, flyover, forest, game space, parking space, river, sparse residential*, and *storage cisterns*.

For PSC, GID is used. It is further expanded to simultaneously implement pixel segmentation and scene classification tasks by adding a scene label to each image. Concretely, GID is used to construct two PSC datasets: PSC5 and PSC15. The PSC5 dataset contains five coarse categories and is constructed from GID images annotated with five categories. The PSC15 dataset contains 15 fine categories and is constructed from GID images annotated with 15 categories. Each image in the PSC datasets contains an image-level scene label and pixel-level land-cover label. Concretely, the PSC dataset is constructed by cropping the original images in GID into small subimages and adding scene category to each subimage. The subimage is set to a size of 224 × 224 pixels. Every subimage is regularized to belong to a certain scene category by ensuring that the number of pixel labels in this category is more than half. A schematic of constructing the multitask dataset is shown in Fig. 3. The detailed information about PSC5 and PSC15 are shown in Figs. 4 and 5, respectively.

3) Implementation Details: For the MSC scenario, the multiple task images are taken as input into the shared branch, and

[1]http://vision.ucmerced.edu/datasets/landuse.html
[2]https://captain-whu.github.io/AID/
[3]http://www.escience.cn/people/JunweiHan/NWPU-RESISC45.html
[4]https://captain-whu.github.io/GID/
[5]https://sites.google.com/view/zhouwx/dataset
[6]https://github.com/phelber/eurosat

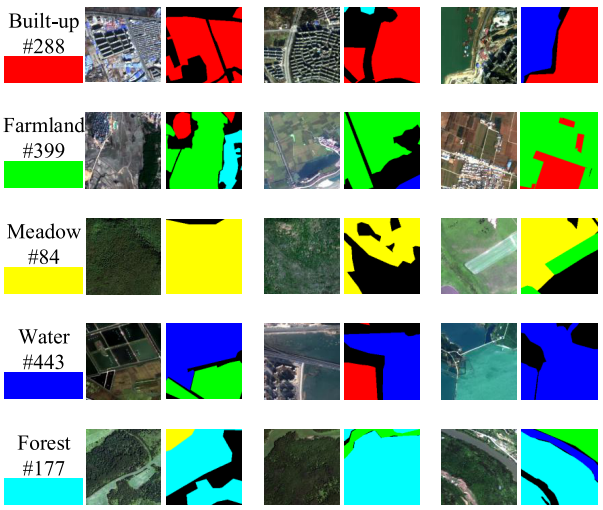

Fig. 4. Example images in PSC5. Each row represents one scene category. (Left column) Scene category, the number of images contained in this category, and the corresponding pixel-level annotation color. Each category displays three groups of images, including the input images and the corresponding ground-truth segmentation maps.

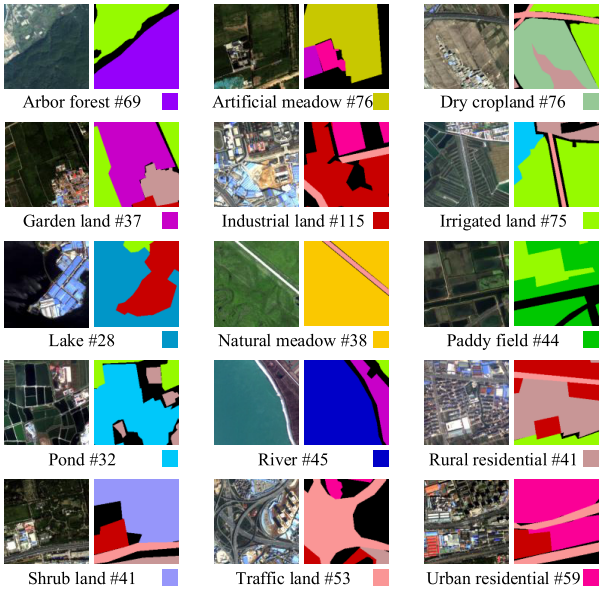

Fig. 5. Example images associated with the 15 scene classes in PSC15. Image, segmentation map, the scene name, the number of images, and the color of the pixel-level label for the category are displayed.

then each data flow into the corresponding task-specific branch for further classification. A softmax classifier is assigned to each task to finish scene classification. For the MSC dataset, we exploit the widely used ResNet-34 as the shared branch for experiments. For the mini-MSC dataset, we exploit WRN-28-4 for its efficiency and lightweight, considering the small number of images in the mini-MSC dataset. WRN-28-4 refers to a wide residual network with layers of 28 and a widening factor of 4 on the convolutional channel number.

For PSC, a classification network is needed for scene classification, and a segmentation network is needed for pixel classification. We exploit state-of-the-art SegNet for pixel classification, which exploits VGG-16 as the encoder network.

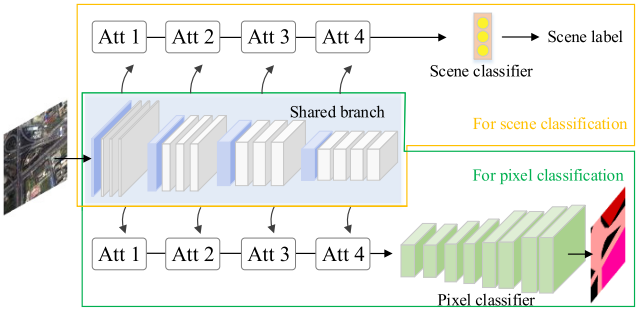

Fig. 6. Network structure of implementing PSC. "Att" represents the attention module. The components surrounded by the yellow box are used for scene classification. The components surrounded by the green box are used for pixel classification. The shared branch is shared by two tasks.

Therefore, VGG-16 is used as the shared branch. A decoder network symmetric to the VGG-16 is used as the pixel classifier. A softmax classifier is used as the scene classifier. A schematic is shown in Fig. 6.

To build the attention module, two convolutional layers of 1×1 kernels together with nonlinear activation layers are stacked. Each attention module is latched onto one convolutional block of the shared branch. To build the extractor, a convolutional layer with a filter size of 3×3 following a pooling layer is used. Each extractor is inserted between two attention modules for passing information from one module to another. The detailed components of the proposed method and the number of parameters are shown in Table II. "Attention modules" represents the parameters of attention modules needed for one task. "Extractors" represents the parameters of all extractors which are shared among multiple tasks.

In the MSC scenario, all input images from different tasks are resized to 224×224 pixels, and the batch size is set to 32. In the PSC scenario, the input image size is set to 224×224 pixels with a batch size of 2. In each dataset, 80% of the images are used as the training set and the remaining images are used as the test set. The MTLN is trained from scratch. Adam optimizer is used. The learning rate is set to 10^{-4}. All the experiments are performed on an Intel Core i7-5930 K workstation with GeForce GTX Titan X GPU, 3.50-GHz CPU, and 64-G RAM.

4) Evaluation Metrics: The experimental results are assessed with classification accuracy and the number of network parameters. For scene classification, the overall accuracy (OA) is used. For the pixel-level semantic segmentation, the accuracy is evaluated on all pixels of the test image. Two metrics-mean intersection over union (mIoU) and pixel accuracy (PA)-are used.

Let p_{ij} denote the number of pixels of class i predicted to belong to class j. $C+1$ represents the number of categories, which contains C defined categories and one unknown category

$$\text{mIoU} = \frac{1}{C+1} \sum_{i=0}^{C} \frac{p_{ii}}{\sum_{j=0}^{C} p_{ij} + \sum_{j=0}^{C} p_{ji} - p_{ii}}$$

$$\text{PA} = \frac{\sum_{i=0}^{C} p_{ii}}{\sum_{i=0}^{C} \sum_{j=0}^{C} p_{ij}}.$$

TABLE II

NUMBER OF PARAMETERS FOR EACH COMPONENT OF MTLN. THE PARAMETER DIFFERENCES BETWEEN PSC5 AND PSC15 ARE IGNORED. THE PARAMETERS OF THE SCENE CLASSIFIER ARE ALSO IGNORED

Parameters	Shared branch	Scene classifier	Pixel classifier	Attention modules	Extractors	Total
MSC (ResNet-34)	21.29M	∼0M	-	1.06M / task	1.58M	26.11M
mini-MSC (WRN-28-4)	5.86M	∼0M	-	0.31M / task	0.83M	7.62M
PSC (VGG-16)	15.49M	∼0M	9.44M	1.80M / task	6.27M	34.80M

TABLE III

EXPERIMENTAL RESULTS OF OA (%) AND NUMBER OF PARAMETERS ON THE MSC. (a) RESULTS ON MSC DATASET WITH RESNET-34 AS BACKBONE. (b) RESULTS ON MINI-MSC DATASET WITH WRN-28-4 AS BACKBONE

(a)

Dataset	Baseline	Fine-tune	Cross-Stitch [47]	DEN [39]	Uncer. Weig. [48]	MTLN(Ours)
Merced	92.66	95.46	77.88	95.57	89.18	**97.66**
AID	88.42	88.61	53.85	89.84	86.54	**92.54**
NWPU	86.31	87.21	31.49	88.27	75.24	**89.71**
Average	89.13	90.43	54.40	91.23	83.65	**93.30**
No. parameters	63.92M	63.92M	63.92M	31.29M	21.34M	26.11M

(b)

Dataset	Baseline	Fine-tune	Cross-Stitch [47]	DEN [39]	Uncer. Weig. [48]	MTLN(Ours)
Merced	90.84	91.19	91.88	90.62	90.14	**92.17**
AID	82.05	82.43	83.20	84.38	82.69	**85.31**
NWPU	74.20	77.62	75.78	77.60	80.29	**82.55**
Average	82.36	83.74	83.62	84.20	84.37	**86.68**
No. parameters	17.58M	17.58M	17.58M	7.72M	5.87M	7.62M

The values of OA, mIoU, and PA are in the range of 0–1. The larger values indicate better performance.

B. MSC

1) Compared Methods: For the MSC scenario, a comparison with single-task methods and multitask state-of-the-arts is conducted.

First, to validate whether the proposed method outperforms the fine-tuned method, a comparison with baseline and fine-tune is conducted. The baseline uses three individual scene classification networks to finish prediction tasks. Each network is trained from scratch independently. The number of parameters is approximately three times the shared branch with a minor difference because of the scene classifier. Fine-tune shares the same architecture with the baseline, but the network is pretrained on the ImageNet and further fine-tuned on the target remote sensing dataset.

Second, the proposed method is compared with three state-of-the-art MTL methods: cross-stitch network (Cross-Stitch) [47], deep elastic network (DEN) [39], and uncertainty weights network (Uncer. Weig) [48]. Cross-stitch network uses multiple independent networks to implement multiple tasks with each network assigned to one task. At the same time, to share information between networks, a cross-stitch unit is used to connect the features of the same layer in different networks. The number of parameters is the same as the baseline. DEN performs MTL by dynamically selecting a model from multiple candidate models. A selection strategy is developed to determine which model is used. Uncertainty weights network assigns a shared branch for multiple tasks for feature extraction and splits it into multiple classifier branches at the classifier layer for the final prediction of each specific task. Different weights are assigned to the multiple losses, so that different tasks can be optimized in balance. The weight is learned adaptively based on the homoscedastic uncertainty of each task. The number of parameters is approximate to the parameters of the shared branch.

2) Results: Two groups of experiments are designed to evaluate the proposed method. The first group exploits ResNet-34 as the backbone, and experiments are conducted on the MSC dataset. The second group exploits WRN-28-4 as the backbone, and experiments are conducted on the mini-MSC dataset. The performances are evaluated with respect to OA and the number of parameters, and the results are shown in Table III.

The proposed method achieves an average OA of 93.30% with parameters of 26.11M in the first group of experiments and an average OA of 86.68% with parameters of 7.62M in the second group. Our method outperforms all the comparison methods including fine-tune and state-of-the-art MTL methods, which validates the effectiveness of the proposed methods.

TABLE IV
EXPERIMENTAL RESULTS ON THE PSC. THE NUMBER OF PARAMETERS, MIOU (%), PA (%), AND OA (%) ARE EVALUATED. VGG-16 IS EXPLOITED AS THE BACKBONE. (A) RESULTS ON PSC5 DATASET. (B) RESULTS ON PSC15 DATASET

(a)

Method	Segmentation		Classification	No. params
	mIoU	PA	OA	
Baseline	18.97	59.89	88.75	41.32M
Cross-Stitch [47]	29.51	75.77	**95.42**	41.32M
Uncer. Weig. [48]	32.87	76.38	94.58	24.93M
MTLN(Ours)	**34.25**	**76.52**	95.00	34.80M

(b)

Method	Segmentation		Classification	No. params
	mIoU	PA	OA	
Baseline	8.53	39.35	78.12	41.32M
Cross-Stitch [47]	10.37	54.88	80.63	41.32M
Uncer. Weig. [48]	10.80	49.12	77.50	24.93M
MTLN(Ours)	**11.75**	**56.39**	**81.87**	34.80M

Meanwhile, the number of parameters is also competitive among all methods.

The following observations can also be learned from the results. First, fine-tune outperforms most methods in MSC, but achieves a similar performance with other methods in the mini-MSC dataset. It may result from the different task interferences of the two datasets. In the MSC dataset, the categories to be classified for each task are hugely different, resulting in interference among tasks. The mini-MSC dataset contains the same categories for each task, and the number of images per category is 100, resulting in more related tasks. Fine-tune performs each task individually, which avoids task interference. Such characteristic is effective on tasks with interference yet less important for related tasks. MTLN is further superior to fine-tune, showing that the proposed method can avoid interference and exploit multiple tasks to further promote performance. Second, the cross-stitch network performs poor on the MSC dataset but performs well on the mini-MSC dataset. This is because in the MSC dataset, different tasks have various difficulties. Cross-stitch network is dominated by the easier task (Merced), resulting in an imbalanced training [29]. The mini-MSC dataset has the same categories for each task and the same number of images per category, leading to a relatively balanced loss over each task. This phenomenon also validates the importance of our MTL framework.

C. PSC

We further evaluate the proposed method on the PSC scenario. The experiments are conducted on the PSC5 and PSC15 datasets, respectively. VGG-16 is exploited as the backbone network. The performance is compared with the state-of-the-art MTL methods: cross-stitch network [47] and uncertainty weights network [48]. Table IV summarizes the comparison results with respect to mIoU, PA, and OA.

The baseline uses a VGG-16 to perform scene classification and a SegNet to perform pixel-level semantic segmentation. Thus, the parameter of the baseline is 41.32M, which is the sum of the parameters of two separate networks. The cross-stitch network assigns an independent network for each task to avoid interference between tasks. The network parameters are the same as baseline, which is 41.32M. Uncertainty weights network assigns a shared branch for two tasks for feature extraction and then connects a pixel classifier and a scene classifier for the final prediction of each specific task. Therefore, the parameter of uncertainty weights network is 24.93M.

The mIoU, PA, and OA of the proposed method on the PSC5 dataset are 34.25%, 76.52%, and 95.00%, respectively. A performance of 11.75%, 56.39%, and 81.87% on PSC15, respectively, was obtained. It can be seen that our method achieves a large performance increase compared with the uncertainty weights network, while the number of network parameters only increases moderately. What is more, the performance outperforms the cross-stitch network on most tasks, while a much smaller amount of parameters are needed.

A visualization of the segmentation results and the attention masks on different tasks are also shown in Fig. 7. The attention mask is learned from the last attention module of each task, coming from the same feature channel. It can be seen that the attention masks for the segmentation task and scene classification task are different. It demonstrates that the proposed attention module can attend to different information according to the executed task.

D. Ablation Analysis

This section validates the effectiveness of each component of the proposed method. Experiments are conducted on the MSC dataset with ResNet-34 as the shared branch and the mini-MSC dataset with WRN-28-4 as the shared branch. The results are shown in Table V.

The MTLN consists of the shared branch and multiple task-specific branches. This shared-specific architecture can handle multiple tasks, and the task-specific branch uses multiple attention modules to distill the shared features for a specific task. Three variants are developed.

1) Baseline uses three individual classification networks with each trained separately.
2) We attach attention modules to each individual classification network. This variant performs each task independently, and three networks are used in total. Therefore, the number of parameters is three times the sum of the shared branch, scene classifier, attention modules, and extractors. We denote this variant as baseline + attention.
3) We exploit a shared branch to conduct multiple tasks. Multiple separate classifiers are exploited with each for one task. This variant is denoted as shared-specific architecture.

By comparing baseline to shared-specific architecture, it can be seen that a simple combination of multiple tasks (shared-specific architecture) is inferior when tasks have interference [results in Table V(a)] but effective when tasks are similar [results in Table V(b)]. But exploiting the attention modules (i.e. MTLN with all the components working) achieves the

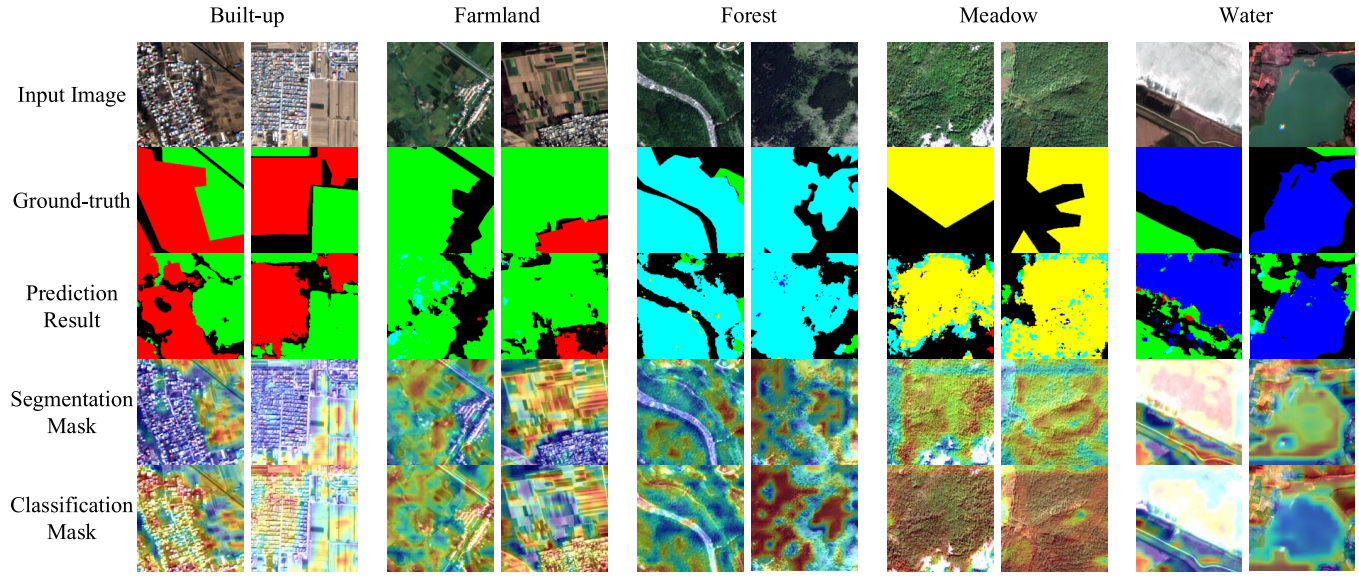

Fig. 7. Semantic segmentation and attention mask results on the PSC5 dataset. Five scene categories are displayed with each category containing two images. First row: input images. Second row: ground-truth semantic segmentation maps corresponding to the images. Third row: predicted segmentation maps from MTLN. Fourth and fifth rows: attention masks for the semantic segmentation task and scene classification task, respectively.

TABLE V

ABLATION RESULTS OF OA (%) AND NUMBER OF PARAMETERS ON MSC. (a) ABLATION ON MSC DATASET. (b) ABLATION ON MINI-MSC DATASET

(a)

Method	Merced	AID	NWPU	Average	No. parameters
Baseline	92.66	88.42	86.31	89.13	63.92M
Baseline+Attention	91.78	88.05	86.68	88.84	71.85M
Shared-Specific Architecture	88.46	77.40	68.03	77.96	21.34M
MTLN (Ours)	**97.66**	**92.54**	**89.71**	**93.30**	26.11M

(b)

Method	Merced	AID	NWPU	Average	No. parameters
Baseline	90.84	82.05	74.20	82.36	17.58M
Baseline+Attention	90.41	81.42	74.72	82.18	21.26M
Shared-Specific Architecture	90.26	82.99	79.88	84.37	5.87M
MTLN (Ours)	**92.17**	**85.31**	**82.55**	**86.68**	7.62M

optimal results. In addition, the results of baseline + attention validate that simple attention is ineffective when used solely, unless combined with the MTL.

E. Generalization Experiments

The proposed method builds a generic model and can quickly adapt to new unseen tasks with the learned general knowledge. To evaluate the generalization ability, experiments are conducted on two new datasets: PatternNet and EuroSAT. Two patterns are considered: using the pretrained MTLN as a feature extractor (abandon the classifiers in MTLN) and fine-tuning MTLN. We implement these patterns and consider several baselines.

1) *Scratch:* A ResNet-34 trained from scratch with the training set of the unseen dataset is used.
2) *Pretrained ResNet-34 on ImageNet:* ResNet-34 pretrained on the ImageNet is used as a feature extractor. A new FC layer specific to the data of interest is assigned by removing the original classification layer. The parameters of the feature extractor are fixed, and the new classifier is trained with the training set.
3) *Pretrained Shared Branch on MSC Dataset (Ours):* The shared branch of MTLN pretrained on the MSC dataset is used as a feature extractor. The task-specific branches are abandoned. A new FC classifier is attached at the end of the shared branch to implement the scene classification. Similarly, the parameters of the feature extractor are fixed, and the new classifier is trained with the training set.
4) *Pretrained ResNet-34 on ImageNet + Fine-Tuning:* ResNet-34 is pretrained on ImageNet and then fine-tuned on the training set of the unseen dataset, in which the final classification layer is adjusted for each specific task.
5) *Pretrain MTLN + Fine-Tuning (Ours):* The MTLN pretrained on the MSC dataset is exploited and a new

TABLE VI
GENERALIZATION COMPARISON WITH STATE-OF-THE-ARTS ON UNSEEN DATASETS. THE OA (%) IS EVALUATED

Dataset	Scratch	Pre-trained ResNet-34 on ImageNet	Pre-trained shared branch on MSC dataset (Ours)	Pre-trained ResNet-34 on ImageNet + Fine-tuning	Pre-train MTLN + Fine-tuning (Ours)
PatternNet	92.53	89.17	93.12	94.15	**94.17**
EuroSAT	80.24	82.75	85.03	**87.86**	86.35

task-specific branch is added to cope with the new dataset. The parameters of the shared branch are fixed. The parameters of the new task-specific branch are initialized from a pretrained task-specific branch (Merced branch in experiments) and then fine-tuned with the training set of the unseen dataset.

The experimental results are shown in Table VI. In each unseen dataset, the training set which contains 100 images per category is used to learn the new classifier or fine-tune, and the remaining images are used for the test. It is observed that our method achieves comparable performance, which verifies the generalization ability of the proposed method.

V. CONCLUSION

This article presents a generalized scene classification model by adopting multiple small-scale remote sensing datasets. An efficient MTLN is proposed to explore complementary information among multiple tasks and distill needed features for each specific task. The proposed method outperforms networks which are pretrained on ImageNet and fine-tuned on a target remote sensing dataset. Furthermore, the method outperforms state-of-the-art methods with a small number of parameters. In the future, we will consider multiscale strategies to deal with images of different sizes and resolutions, so that a generalizable model can be learned. In addition, we will explore the performance of MTLN on other combinations of datasets and tasks. To realize concrete applications, an out-of-the-box model will be considered so that the learned generalized model can be directly applied to different new tasks.

REFERENCES

[1] 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.

[2] X. Zheng, Y. Yuan, and X. Lu, "A deep scene representation for aerial scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4799–4809, Jul. 2019.

[3] X.-Y. Tong *et al.*, "Land-cover classification with high-resolution remote sensing images using transferable deep models," *Remote Sens. Environ.*, vol. 237, Feb. 2020, Art. no. 111322.

[4] Z. Yang, H. Yu, M. Feng, W. Sun, and X. Lin, "Small object augmentation of urban scenes for real-time semantic segmentation," *IEEE Trans. Image Process.*, vol. 29, pp. 5175–5190, 2020.

[5] Y. S. Liu, Z. Han, C. Chen, L. Ding, and Y. Liu, "Eagle-eyed multitask CNNs for aerial image retrieval and scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 9, pp. 6699–6721, Sep. 2020.

[6] X. Zheng, B. Wang, X. Du, and X. Lu, "Mutual attention inception network for remote sensing visual question answering," *IEEE Trans. Geosci. Remote Sens.*, early access, May 31, 2021, doi: 10.1109/TGRS.2021.3079918.

[7] L. Zhang, M. Wang, R. Hong, B. Yin, and X. Li, "Large-scale aerial image categorization using a multitask topological codebook," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 535–545, Feb. 2016.

[8] X. Lu, T. Gong, and X. Zheng, "Multisource compensation network for remote sensing cross-domain scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 4, pp. 2504–2515, Apr. 2020.

[9] Q. Bi, K. Qin, Z. Li, H. Zhang, K. Xu, and G.-S. Xia, "A multiple-instance densely-connected ConvNet for aerial scene classification," *IEEE Trans. Image Process.*, vol. 29, pp. 4911–4926, 2020.

[10] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Convolutional neural networks for large-scale remote-sensing image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 2, pp. 645–657, Feb. 2017.

[11] W. Yang, X. Yin, and G.-S. Xia, "Learning high-level features for satellite image classification with limited labeled samples," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 8, pp. 4472–4482, Aug. 2015.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.

[13] D. Marmanis, M. Datcu, T. Esch, and U. Stilla, "Deep learning Earth observation classification using ImageNet pretrained networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 1, pp. 105–109, Jan. 2016.

[14] B. Zhao, B. Huang, and Y. Zhong, "Transfer learning with fully pretrained deep convolution networks for land-use classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 9, pp. 1436–1440, Sep. 2017.

[15] F. Hu, G. Xia, J. Hu, and L. Zhang, "Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery," *Remote Sens.*, vol. 7, pp. 14680–14707, Nov. 2015.

[16] K. Nogueira, O. A. B. Penatti, and J. A. dos Santos, "Towards better exploiting convolutional neural networks for remote sensing scene classification," *Pattern Recognit.*, vol. 61, pp. 539–556, Jan. 2017.

[17] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *Proc. CVPR*, Jun. 2011, pp. 1521–1528.

[18] L. Zhang, X. Wan, and B. Sun, "Tropical natural forest classification using time-series Sentinel-1 and Landsat-8 images in Hainan Island," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2019, pp. 6732–6735.

[19] L. Ru, B. Du, and C. Wu, "Multi-temporal scene classification and scene change detection with correlation based fusion," *IEEE Trans. Image Process.*, vol. 30, pp. 1382–1394, 2021.

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

[21] 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.

[22] T. Gong, X. Zheng, and X. Lu, "Cross-domain scene classification by integrating multiple incomplete sources," *IEEE Trans. Geosci. Remote Sens.*, early access, May 31, 2021, doi: 10.1109/TGRS.2021.3079918.

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

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, Jan. 2012, pp. 1097–1105.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, May 2015.

[26] O. A. B. Penatti, K. Nogueira, and J. A. dos Santos, "Do deep features generalize from everyday objects to remote sensing and aerial scenes domains?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2015, pp. 44–51.

[27] E. Othman, Y. Bazi, F. Melgani, H. Alhichri, N. Alajlan, and M. Zuair, "Domain adaptation network for cross-scene classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 8, pp. 4441–4456, Aug. 2017.

[28] K. Lyu, Y. Li, and Z. Zhang, "Attention-aware multi-task convolutional neural networks," *IEEE Trans. Image Process.*, vol. 29, pp. 1867–1878, 2020.

[29] S. Liu, E. Johns, and A. J. Davison, "End-to-end multi-task learning with attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 1871–1880.
[30] L. Liebel, K. Bittner, and M. Körner, "A generalized multi-task learning approach to stereo DSM filtering in urban areas," *ISPRS J. Photogramm. Remote Sens.*, vol. 166, pp. 213–227, Aug. 2020.
[31] M. Long, Z. Cao, J. Wang, and P. Y. Yu, "Learning multiple tasks with multilinear relationship networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Nov. 2017, pp. 1594–1603.
[32] T. Gong, X. Zheng, and X. Lu, "Remote sensing scene classification with multi-task learning," in *Proc. China High Resolution Earth Observ. Conf.*, 2020.
[33] C. Qiu, L. Liebel, L. H. Hughes, M. Schmitt, M. Korner, and X. X. Zhu, "Multitask learning for human settlement extent regression and local climate zone classification," *IEEE Geosci. Remote Sens. Lett.*, early access, Nov. 24, 2020, doi: 10.1109/LGRS.2020.3037246.
[34] H. Guo, Q. Shi, B. Du, L. Zhang, D. Wang, and H. Ding, "Scene-driven multitask parallel attention network for building extraction in high-resolution remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 5, pp. 4287–4306, May 2021.
[35] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3712–3722.
[36] R. Hang, F. Zhou, Q. Liu, and P. Ghamisi, "Classification of hyperspectral images via multitask generative adversarial networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 2, pp. 1424–1436, Feb. 2021.
[37] S. Liu, Q. Shi, and L. Zhang, "Few-shot hyperspectral image classification with unknown classes using multitask deep learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 6, pp. 5085–5102, Jun. 2021.
[38] Q. Liu, X. Xiang, Z. Yang, Y. Hu, and Y. Hong, "Arbitrary direction ship detection in remote-sensing images based on multitask learning and multiregion feature fusion," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 2, pp. 1553–1564, Feb. 2021.
[39] C. Ahn, E. Kim, and S. Oh, "Deep elastic networks with model selection for multi-task learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6528–6537.
[40] C. Zhang *e al.*, "Joint deep learning for land cover and land use classification," *Remote Sens. Environ.*, vol. 221, pp. 173–187, Feb. 2019.
[41] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 8, pp. 2011–2023, Aug. 2020.
[42] K.-K. Maninis, I. Radosavovic, and I. Kokkinos, "Attentive single-tasking of multiple tasks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1851–1860.
[43] X. Sun, R. Panda, R. Feris, and K. Saenko, "Adashare: Learning what to share for efficient deep multi-task learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, Nov. 2020, pp. 8728–8740.
[44] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder–decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[45] W. Zhou, S. Newsam, C. Li, and Z. Shao, "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 197–209, Nov. 2018.
[46] P. Helber, B. Bischke, A. Dengel, and D. Borth, "EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Ses.*, vol. 12, no. 7, pp. 2217–2226, Jul. 2019.
[47] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3994–4003.
[48] R. Cipolla, Y. Gal, and A. Kendall, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7482–7491.

Xiangtao Zheng (Member, IEEE) received the M.Sc. and Ph.D. degrees in signal and information processing from the Chinese Academy of Sciences, Xi'an, China, in 2014 and 2017, respectively.

He is currently an Associate Professor with the Key Laboratory of Spectral Imaging Technology CAS, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences. His main research interests include computer vision and pattern recognition.

Tengfei Gong is currently pursuing the Ph.D. degree with the Key Laboratory of Spectral Imaging Technology CAS, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, Shaanxi, China, and also with the University of Chinese Academy of Sciences, Beijing, China.

Her research interests include pattern recognition, computer vision, and machine learning.

Xiaobin Li is with Beijing Institute of Remote Sensing Information, Beijing, China. His research interests include computer vision and pattern recognition.

Xiaoqiang Lu (Senior Member, IEEE) is currently a Full Professor with the Key Laboratory of Spectral Imaging Technology CAS, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, Shaanxi, China. His research interests include pattern recognition, machine learning, hyperspectral image analysis, cellular automata, and medical imaging.