A Learnable Joint Spatial and Spectral Transformation for High Resolution Remote Sensing Image Retrieval

Yameng Wang, Shunping Ji, *Member, IEEE*, and Yongjun Zhang

***Abstract*—Geometric and spectral distortions of remote sensing images are key obstacles for deep learning-based supervised classification and retrieval, which are worsened by cross-dataset applications. A learnable geometric transformation model imbedded in a deep learning model has been used as a tool for handling geometric distortions to process close-range images with different view angles. However, a learnable spectral transformation model, which is more noteworthy in remote image processing, has not yet been designed and explored up to now. In this paper, we propose a learnable joint spatial and spectral transformation (JSST) model for remote sensing image retrieval (RSIR), which is composed of three modules: a parameter generation network (PGN); a spatial conversion module; and a spectral conversion module. The PGN adaptively learns the geometric and spectral transformation parameters simultaneously from the different input image content, and these parameters then guide the spatial and spectral conversions to produce a new modified image with geometric and spectral correction. Our learnable JSST is imbedded in the front-end of the deep-learning-based retrieval network. The spatial and spectral-modified inputs provided by the JSST endow the retrieval network with better generalization and adaptation ability for cross-dataset RSIR. Our experiments on four open-source RSIR datasets confirmed that our proposed JSST embedded retrieval network outperformed state-of-the-art approaches comprehensively.**

***Index Terms*—Convolutional neural network (CNN), remote sensing image retrieval (RSIR), spatial transformation, spectral transformation.**

I. Introduction

IMAGE retrieval technology makes it possible to retrieve a certain number of ranked images from an image dataset or the internet according to their degree of similarity to a query image or keyword. Remote sensing image retrieval (RSIR), which searches related or similar images from a remote sensing image set, can be categorized into text-based RSIR and content-based RSIR. The former finds labeled images from a dataset according to query keywords or labels [1]–[3] and often requires an early labor-intensive data annotation process to label each image in the search dataset. The latter searches images according to the similarity between the contents of the search and query images, which is close to human visual perception and is the current mainstream thought. CBRSIR roughly can be divided into three steps: feature extraction; feature reduction; and similarity calculation. Feature extraction employs a feature extractor, which is usually implemented by a convolutional neural network (CNN), to obtain the abstract feature representation of each image in a remote sensing dataset, all of which then form a feature library. The feature representation of the query image is extracted by the same extractor. Feature reduction, such as pooling or encoding technologies, may not be necessary in some conventional methods that have designed compact features; but in deep learning, the feature reduction is usually required. Similarity calculation then matches the reduced features in the library with the query feature one-by-one using a specific similarity measure. The search images then are ranked according to the similarity score, and the top-ranked images are returned as the retrieval result.

The one aspect of RSIR is to extract distinctive features for representing each category; and over the last few decades, the computer vision and remote sensing communities have continued to innovate feature extraction to improve its feature representation ability. Early feature representation relied on empirical design from a human expert's understanding of the images, which usually was restricted to the simple and intuitive patterns of an image, such as lines, shapes, and textures, and the often-called low-level features, of which SIFT [4], LBP [5], and HOG [6] are well known. These features describe the local patterns of an image that then are aggregated sometimes to form global feature representations, which are called middle-level features. bag-of-words (BoW) [7], vector of locally aggregated descriptors (VLAD) [8], Fisher kernels (FK) [9], and recent efficient match kernels (EMK) [10] and memory vectors [11] are examples of middle-level features. Most of them also function as feature reduction approaches by controlling the feature dimension to be the output. Nowadays, mainstream feature representation technology is based on deep learning. Image features extracted by deep learning methods, especially CNN, are realized with multiple convolutional layers abstraction [12]–[14], which are often called high-level features. These features are reduced with pooling [15], [16] or encoding methods [7]–[11] to form a library of image retrieval. Both traditional and deep learning-based feature extraction methods have been applied to RSIR [17]–[23].

Recently, some studies focused on designing special deep neural network structures. Zhou *et al.* [20] introduced a three-layer

Manuscript received April 9, 2021; revised June 20, 2021 and July 26, 2021; accepted August 3, 2021. Date of publication August 12, 2021; date of current version August 27, 2021. This work was supported by the National Key Research and Development Program of China under Grant 2018YFB0505003. *(Corresponding author: Shunping Ji.)*

The authors are with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China (e-mail: ymw@whu.edu.cn; jishunping@whu.edu.cn; zhangyj@whu.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2021.3103216

perceptron to learn low-dimensional features. Noh *et al.* [24] utilized an attention mechanism to score the relevant features. SENet [25] exploits a channel-weighted attention strategy to realize feature recalibration. SKNet [26] is regarded as a multi-branch version of SENet that can obtain the adaptive receptive field size. NTS-Net [27] uses a self-supervision mechanism to locate the area of interest. SBS-CNN [28] combines a feature learning network and a similarity learning network to predict the similarity percentage between two images directly for unsupervised RSIR task.

Other studies paid more attention to the feature aggregation and reduction aspects of the CNN-based features. Borrowing ideas from sum pooling, Babenko and Lempitsky [29] aggregated deep convolutional features into a compact descriptor by distributing different weights to each pixel within a channel. Kalantidis *et al.* [30] added the influence of channels to pixels on the basis of SpoC. In [31], the outputs of the convolutional layer were encoded by VLAD, which generated the column features for similarity matching. Wang *et al.* [23] generated distinctive and compact features by introducing bilinear pooling and a channel and spatial attention module. Radoi and Datcu [32] introduced error-correcting output codes encoding and decoding to translate multilabel classification to binary classification.

However, the other aspect, i.e., how to preprocess the original remote sensing images for better retrieval performance, has not been deeply investigated. It is crystal clear that the performance of a retrieval model is profoundly affected by geometric and spectral distortions in a remote sensing image, especially in cross-dataset applications. A robust RSIR model that addresses geometric and radiometric distortion, caused by different view angles, dynamic atmosphere, sensors, under- or over-exposure, etc., is greatly needed but is still lacking in current deep learning-based retrieval methods. The widely used data augmentation approach in machine learning has been introduced to RSIR, but it is empirically performed outside deep learning models through the production of the redundant counterparts of the original input, instead of end-to-end learning. In fact, a learnable geometric or spectral transformation approach has its distinct advantage in image retrieval. As a basic understanding, RSIR is highly affected by the content of the current remote image. A rigid transformation with fixed or empirical parameters hardly adapts to various situations. A learnable geometric or spectral transformation module imbedded in a deep learning network can better understand the content of the current image and generate a proper image from both of the geometric and spectral views for the image retrieval task in the same training/prediction loop.

There are several methods in the machine learning community that have geometric transformation imbedded in their learning-based models without an increase in the amount of training data. Kosiorek *et al.* [33] achieved viewing angle independent reconstruction from the part capsule to the object capsule considering the geometric relationship within an object. By adding deformation parameters in the convolution layers and ROI pooling layers, a deformable convolutional network [34] changed the range of the receptive field to adapt to the geometric deformation of objects. Esteves *et al.* [35] made use of the so-called "polar origin predictor" and "polar transformer" to achieve rotation and scale invariance. Jaderberg *et al.* [36] performed spatial transform on CNN feature maps to improve their recognition ability on the MNIST dataset. There are also studies that attempt to boost geometric robustness with aggregating features extracted from different features [37] or different spatial scales [38], [39].

Unfortunately, there are currently no learning-based spectral transformation approaches that were designed for spectral distortions to the best of our knowledge, although the spectral transformation may be more important in RSIR. Instead, histogram equalization [40], gamma correction [41], and Wallis transformation [42] are widely used, all of them are conventional empirical-based spectral transformation methods, and the former two are nonlinear-based methods and the latter is linear-based. Obviously, the learning-based integrated geometric and spectral transformation approaches simply do not yet exist in either computer vision or remote sensing community.

To tackle this problem, we propose a compact plug-in anti-distortion model within an end-to-end learning scheme specifically designed for RSIR, called the joint spatial and spectral transformation (JSST) model. It adaptively learns different parameters from the specific content of an image and generates a revised image with geometric and spectral correction. It consists of three parts: a parameter generation network (PGN); a spatial conversion module; and a spectral conversion module. The PGN adaptively learns spatial and spectral transformation parameters simultaneously and guides the two conversions to output a revised image with the repaired spatial and spectral distortions, thereby increasing the robustness of the retrieval network.

The proposed JSST integrates conventional distortion correction strategies into a deep learning framework and automatically adjusts correction parameters. JSST fills the blank of lacking learning-based spectral transformation approaches and learns unique correction parameters for each input image instead of applying empirical parameters for all images. Furthermore, our method combines spatial and spectral correction as a learnable module embedded in an end-to-end network for the first time.

The main contributions of this article are as follows.

1) The first learnable spatial and spectral transformation model. Our JSST is designed for correcting the possible spatial and spectral distortions of an image for the following RSIR task. It is fully imbedded in a deep learning retrieval model and adaptively learns parameters according to the contents of an image and performs spatial and spectral transformations on the input image simultaneously. It is totally different from empirical data augmentation executed outside a learning procedure.
2) The conciseness and interpretability of our JSST. Our JSST is expressed with explicit mathematical models; and under loose assumptions, it can simulate most of the spatial and special distortions. Moreover, the mathematical models used in our JSST can be easily replaced with alternative models chosen by the users.
3) The high performance of the JSST. With the same baseline retrieval network, JSST significantly outperformed other recent image retrieval methods and reached the state-of-the-art methods in different open-source datasets. This achievement highlights the importance of learning-based image preprocessing in RSIR with respect to the

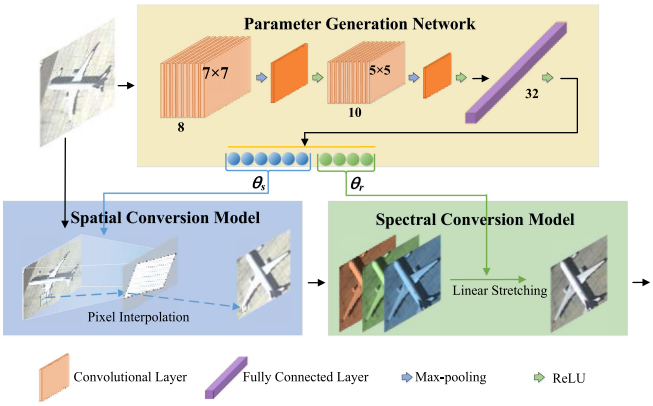


Fig. 1. Proposed JSST for image retrieval.

mainstream studies of merely concentrating on network structure adjustment and feature reduction.

Section II provides a detailed description of our JSST module; our extensive experiments to demonstrate the effectiveness of JSST are presented in Section III; further general discussion is offered in Section IV; and our conclusions are provided in Section V.

II. Joint Spatial and Spectral Transformation Model

In Section II-A, we describe our JSST and its three core modules: the PGN; the spatial conversion module; and the spectral conversion module. Then, in Section II-B, we show how it is integrated it into a complete RSIR network that was specifically designed for cross-dataset image retrieval.

A. Joint Spatial and Spectral Transformation

Our JSST, which is shown in Fig. 1, automatically learns the proper spatial and spectral parameters to convert an original input image into another image for emphasizing the objects/scenes of interest and achieving better spectral and visual presentation. Its goal is to better describe the content of the image so that it is more easily discriminated by an image retrieval network. The PGN module outputs two sets of parameters simultaneously, each of which guides the subsequent spatial conversion and spectral conversion modules. The spectral conversion module is executed on the resampled image from the spatial conversion module, which outputs enhanced image for the image retrieval process.

1) Parameter Generation Network: The PGN adaptively learns the geometric and spectral distortion parameters of each image in a deep learning model, which is image-customized (i.e., each input image passing through the PGN corresponds to a unique set of conversion parameters) and conceptually-different from the commonly-used data augmentation approach that resamples the image with uniform and empirical parameters outside a deep learning model. The PGN is mathematically denoted as a nonlinear function $f(\cdot)$, with an input image $U_0 \in R^{C\times H\times W}$, where C, H, and W are the number of spectral channels, image height, and image width, respectively. Through $f(\cdot)$, the input image is transformed into a set of parameters denoted as θ, which is the combination of spatial conversion parameter θ_s and spectral conversion parameter θ_r in this work, as shown in

$$\theta = f(U_0) \text{ where } \theta = \{\theta_s, \theta_r\}. \tag{1}$$

In this article, $f(\cdot)$ is implemented by a light convolutional network with a series of convolutional layers, pooling layers, a fully connected (FC) layer, and a last regression layer (see Fig. 1). The convolutional layers extract the geometric and spectral features from the image, and the FC layer and regression layer compress them into the parameters θ with fixed numbers according to the defined conversion models.

2) Spatial Conversion Model: The spatial conversion module is designed to highlight the objects or scenes of interest in the image from a new viewing angle. In this work, it is implemented through a warp module, which consists of two steps: a pixel coordinate conversion and a pixel interpolation.

Suppose the transformed regular grid is G^s and the original grid is G^0, which share the same image size. For an arbitrary point indexed i, its transformed coordinate is $X_i^s = (x_i^s, y_i^s) \in G^s$ and the corresponding original coordinate is $X_i^0 = (x_i^0, y_i^0) \in G^0$. The mapping relationship between the original image coordinates and the transformed image coordinates is as follows:

$$X_i^0 = \Gamma_{\theta_s}(X_i^s). \tag{2}$$

The symbol Γ represents the spatial conversion function, with the corresponding spatial conversion parameters θ_s. It has been proven that an eight-parameter perspective projection can be well approximated by a six-parameter affine projection in a small local region of an image [43], and a push-broom imaging model can be simulated as the perspective model within a small region [44]. Therefore, we can reasonably apply six-parameter affine transformation, as shown in (3), because the remote sensing images have been cropped into small patches before inputting them into a retrieval model

$$\begin{pmatrix} x_i^0 \\ y_i^0 \end{pmatrix} = A_\theta \begin{pmatrix} x_i^s \\ y_i^s \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{s1} & \theta_{s2} & \theta_{s3} \\ \theta_{s4} & \theta_{s5} & \theta_{s6} \end{bmatrix} \begin{pmatrix} x_i^s \\ y_i^s \\ 1 \end{pmatrix} \tag{3}$$

where A_θ represents the six-parameter affine projection. θ_{si} (i = 12, …6) are six scalar elements.

A pixel-based interpolation function Ψ then translates the original image to a new image according to the affine parameters. In order to realize the end-to-end training strategy, Ψ must be derivable for the iterative back propagation of the network. We chose bilinear interpolation [45] as it is differentiable.

The whole warp module can be mathematically written as

$$U_s = \Psi(\Gamma_{\theta s}(G^s)). \tag{4}$$

The functions Γ and Ψ can have other implementations. For example, thin plate spline (TPS) [46] is another widely-used geometric transformation with nonlinear fitting ability besides the rigid affine and perspective projection. As for pixel-based interpolation, high order polynomials, such as bicubic interpolation [47] are options.

3) Spectral Conversion Model: We established the spectral conversion model according to two empirical observations: first, between-pixel calibration in a remote sensing image patch is not necessary because the patch is typically very small (e.g.,

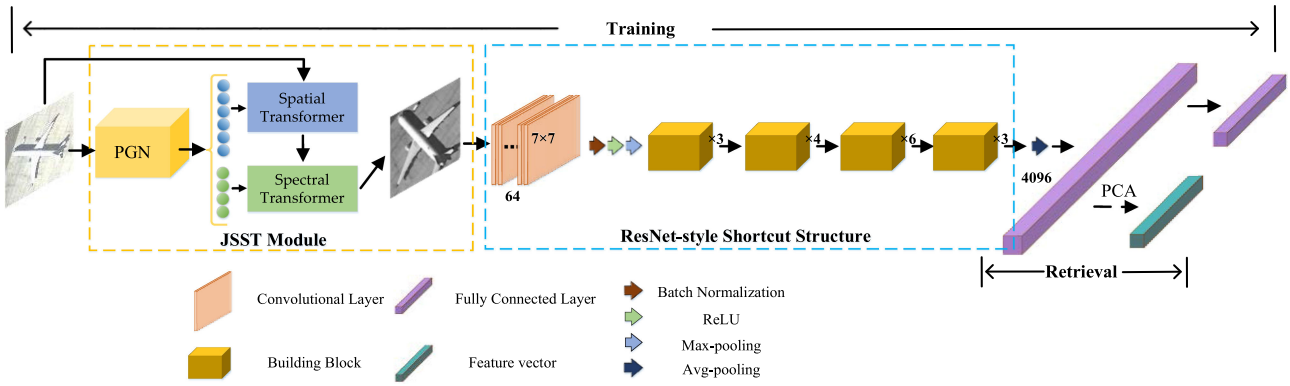


Fig. 2. Structure of the proposed retrieval network with imbedded JSST. ResNet-34 is used for feature abstraction. The output vector is obtained from the FC layer in training stage, and from PCA in retrieval stage for better generalization ability.

512×512 pixels); second, the overall spectral difference between patches is the main factor. Hence, we only need to adjust the linear and independent changes of each spectral channel to simulate almost all the radiometric variations, including illumination change, underexposure or overexposure, and color cast, of an image patch. The brightness ratio among the spectral channels is then parameterized. Four parameters $\theta_r = \{\theta_{r1},\theta_{r2},\theta_{r3},\theta_{r4}\}$, the first three for linear stretching and the last one for shared translation bias, are constructed as follows:

$$U_{sri} = \theta_{ri}U_{si} + \theta_{r4}, i \in [1,3] \tag{5}$$

where U_{sr} is the converted image and i denotes one of the three spectral channels. If multispectral images are used, the spectral channel number will be set accordingly.

According to Fig. 1, the spectral conversion is applied on the spatially converted image. We will discuss the order of execution of the two conversions in Section III-D.

B. JSST-Imbedded Image Retrieval Network

Fig. 2 shows the complete RSIR network with the integrated JSST. The input image is processed by the PGN, which then outputs two sets of parameters to guide the spatial and spectral conversion modules. The output image of JSST is then processed by a series of plain convolutions and a series of ResNet-style [48] shortcut blocks (see in Fig. 2). The convolutions and shortcut blocks were pretrained on the ImageNet [49]. The last feature is compressed by a FC layer FC1 to output a one-dimensional (1-D) vector with 4096 units. In the training stage, the vector is further compressed by another FC2 to obtain a vector with the same length of category number. In the retrieval stage, FC2 is replaced by a principal component analysis (PCA) operation. As FC2 is highly related to the training dataset, PCA can lead to a better generalization ability, which is highly required in the cross-dataset retrieval task. The performance of FC and PCA are discussed in Section III-C.

III. Experiment and Result

A. Data

Six independent open-source RSIR datasets were prepared for our comprehensive evaluation of the effect of our proposed JSST. Among them, only the PatternNet [50] dataset was used to train our network and for comparing with the other methods. The trained CNN models were directly applied to three frequently used RSIS datasets: WHU-RS19 [51], UC Merced Land-Use Dataset (UCM) [52], and RSSCN [53] and two related and more complicated ones: aerial image dataset (AID) [54], and NWPU-RESISC45 (NWPU) [55] for the evaluation process.

PatternNet [50] contains 38 categories of remote sensing scenes or ground objects (airplane, baseball field, basketball court, beach, bridge, cemetery, chaparral, Christmas tree farm, closed road, coastal mansion, crosswalk, dense residential, ferry terminal, football field, forest, freeway, golf course, harbor, intersection, mobile home park, nursing home, oil gas field, oil well, overpass, parking lot, parking space, railway, river, runway, runway marking, shipping yard, solar panel, sparse residential, storage tank, swimming pool, tennis court, transformer station, and wastewater treatment plant). Each category is composed of 800 256×256 high-resolution images selected from Google Earth.

RS19 [51] covers 19 types of scenes (airport, beach, bridge, commercial, desert, farmland, football field, forest, industrial, meadow, mountain, park, parking, pond, port, railway station, residential, river, and viaduct), which were acquired from Google satellite imagery for scene classification or retrieval. The dataset has a totally of 1005 600×600 images with approximately 50 images for each category.

UCM [52] contains 21 categories (agricultural, airplane, baseball diamond, beach, buildings, chaparral, dense residential, forest, freeway, golf course, harbor, intersection, medium residential, mobile home park, overpass, parking lot, river, runway, sparse residential, storage tanks, and tennis court) and has a total of 2100 256×256 remote sensing images from the USGS National Map Urban Area Imagery series.

RSSCN7 [53] consists of seven typical scene categories (field, forest, grass, industry, parking, resident, and rivers-lakes) and has a total of 2800 400×400 diverse remote sensing images that were taken under different seasons and weather changes and were sampled in four different scales.

AID [54] contains 30 categories (airport, bare land, baseball field, beach, bridge, center, church, commercial, dense residential, desert, farmland, forest, industrial, meadow, medium residential, mountain, park, parking, playground, pond, port,

railway station, resort, river, school, sparse residential, square, stadium, storage tanks, and viaduct) and a total of 10000 600×600 aerial scene images.

NWPU [55] is a scene classification dataset containing 45 classes (airplane, airport, baseball diamond, basketball court, beach, bridge, chaparral, church, circular farmland, cloud, commercial area, dense residential, desert, forest, freeway, golf course, ground track field, harbor, industrial area, intersection, island, lake, meadow, medium residential, mobile home park, mountain, overpass, palace, parking lot, railway, railway station, rectangular farmland, river, roundabout, runway, sea ice, ship, snowberg, sparse residential, stadium, storage tank, tennis court, terrace, thermal power station, and wetland) and a total of 31 500 images. Each class consists of 700 256×256 images.

B. Experimental Setup

To compare the performance of our proposed JSST to other state of the art retrieval methods under the same baseline, we used the classic ResNet structure [48] with 34 trainable layers (ResNet34) as the backbone network. ResNet-34 was pretrained on the ImageNet dataset [49] to initialize the weights. Then, with the first three shortcut structures of the ResNet-34 architecture frozen, JSST and the other methods performed fine-tuning on the PatternNet dataset [50]. The pretraining and the fine-tuning constituted the training phase. Please note that the trained model on PatternNet is the final model to be applied on the other datasets for retrieval performance evaluation. All the input images were uniformly resized to 224×224 in both the fine-tuning and retrieval phases. The strategy of learning rate decay during the fine-tuning phase was applied. There were 40 epochs in the process of fine-tuning, among which the learning rates of the 1st–15th epochs were 10^{-3}, those of the 16th-30th epochs were 10^{-4}, and those of the last 10 epochs were 10^{-5}. The batch size was set to 64. SGD for gradient descent was applied to all the retrieval methods except compact bilinear pooling (CBP) [23], which used Adam and did not converge when using SGD. All the experiments were completed on a Linux PC equipped with an NVIDIA GeForce RTX 2070 8G GPU and the PyTorch deep learning framework.

In the retrieval phase, the Pearson correlation coefficient was used to perform similarity matching between the features extracted from the query images and all the remaining images in the dataset. We used precision at k (P@k), where k was the pre-set number of returned images in a query, and the mean average precision (mAP) as measures to evaluate the performance of the different methods.

C. Retrieval Result and Analysis

We mainly compared our JSST with three recent excellent classification networks and other four outstanding image retrieval methods. SENet [25] was the champion of ILSVRC 2017 classification task. SKNet [26] was an improvement on the basis of SENet. NTS-Net [27] used a self-supervision mechanism for the fine-grained visual categorization task to locate the area of interest. DFLA [22] combined an attention module with center loss to form a multi-task learning network structure. In LDCNN [20], network in network [57] was introduced into RSIR, followed by a global average pooling to obtain one-dimensional features. Deep hashing network (DHN) [56] generated binary hash code from an FC layer and introduced a double-branch loss to maintain similarity and hash quality. The recent work [23] proposed an attention boosted CBP and demonstrated better than BoW [7], IFK [58] and other classic aggregation methods in RSIR. In addition, some recent Embedding and Hashing based methods are also compared [59]–[62].

1) Overall Evaluation: The retrieval results of the different methods on the three datasets RS19, UCM, and RSSCN are given in Table I. Our JSST achieved the best results on all three datasets and significantly outperformed the baseline method (ResNet34) and the other recent methods. On RS19, JSST exceeded the baseline more than 5% on mAP and 7% on P@5 and exceeded the second best SENet 1.2% on mAP. On UCM, JSST outperformed the baseline and the second best SENet 2% and 1.3% on mAP. On RSSCN, JSST outperformed the baseline 6% and the second best SENet 1.2% on mAP. NTS-Net did not perform well and was even worse than the baseline method with PCA pooling. On all the datasets, PCA reduction performed better than the commonly used FC layer as the latter was highly affected by the training datasets; this observation is consistent with the previous studies [23], [63].

To save time, we compare our method with only the excellent SENet, SKnet and NTS-Net on the extraordinary large datasets AID and NWPU, the results are given in Table II. The conclusion is similar to that of the Table I, that our JSST outperforms all the other methods on all the evaluation indexes.

The retrieval results of the query images from RS19, UCM, and RSSCN retrieval datasets are shown in Fig. 3, where the query images are shown in the first row, followed sequentially by the results of ResNet34(PCA), CBP, SENet, SKNet, and our JSST.

Fig. 3(a) is from the RS19 dataset, and the left query image is labeled with the airport area, which easily can be confused with the commercial area or the industrial area. ResNet made many such mistakes. Our JSST made only one error at the second lowest rank. In the right column, the category of pond is queried, which ResNet34, CBP, and SKNet confused with football fields. Our JSST made a perfect prediction.

Fig. 3(b) is from the UCM dataset, and the left query image is labeled with dense residential area, which is quite complicated to discriminate from other areas containing houses, such as medium residential, sparse residential, mobile home parks, etc. However, our JSST almost perfectly recognized their distinct differences. In contrast, ResNet, CBP, and SENet made many mistakes. The right image is labeled as a freeway; and all five methods returned some overpass scenes, whereas our JSST performed the best

Fig. 3(c) is from the RSSCN dataset, and the left query image is labeled as an industry area. Distinguishing the industry area from parking or residential areas can be a difficult problem, even for humans. Compared with other methods that made at least four wrong predictions, our JSST made only one mistake. The right query image is categorized as a river lake. ResNet, CBP, and SENet mistook grass or forest areas as rivers or lakes more than three times. Our JSST achieved the best performance.

TABLE I
RETRIEVAL RESULTS OF DIFFERENT METHODS ON THE THREE REMOTE SENSING RETRIEVAL DATASETS

Dataset	Methods	encoder/pooling	mAP	P@5	P@10	P@50	P@100
RS19	ResNet34 (baseline)	FC	0.8867	0.8469	0.8087	0.5755	0.3854
	ResNet34	PCA	0.9131	0.8745	0.8291	0.5474	0.3475
	SE [25]	PCA	0.9318	0.9051	0.8816	0.6593	0.4209
	SK [26]	PCA	0.8901	0.8480	0.8015	0.6301	0.4081
	NTS-Net [27]	PCA	0.8994	0.8562	0.8040	0.6187	**0.4758**
	DFLA [22]	PCA	0.6716	0.5531	0.501	0.3496	0.2484
	LDCNN [20]	Max Pooling	0.6633	0.5551	0.5143	0.3811	0.2765
	DHN [56]	Hashing	0.8445	0.6888	0.6031	0.3769	0.2768
	CBP [23]	PCA	0.8951	0.8490	0.7974	0.6026	0.3901
	RiDe [59]	PCA	0.8954	0.8622	0.8250	0.6226	0.4091
	MiLaN [60, 61]	Hashing	0.8415	0.6714	0.6666	0.4207	0.2796
	T-NLNN [62]	PCA	0.8835	0.8327	0.8133	0.5899	0.3948
	JSST (ours)	FC	0.9099	0.8755	0.8413	0.6129	0.4019
	JSST (ours)	PCA	**0.9432**	**0.9194**	**0.8964**	**0.6974**	0.4354
UCM	ResNet34 (baseline)	FC	0.9096	0.8605	0.8162	0.6669	0.5436
	ResNet34	PCA	0.8994	0.8562	0.8040	0.6187	0.4758
	SE [25]	PCA	0.9167	0.8786	0.8419	0.7003	0.5771
	SK [26]	PCA	0.9018	0.8610	0.8167	0.6895	0.5774
	NTS-Net [27]	PCA	0.8352	0.7610	0.6950	0.4560	0.3438
	DFLA [22]	PCA	0.6059	0.491	0.4348	0.309	0.246
	LDCNN [20]	Max Pooling	0.7338	0.6552	0.6221	0.526	0.4407
	DHN [56]	Hashing	0.8905	0.7395	0.6624	0.516	0.4179
	CBP [23]	PCA	0.9056	0.8638	0.8367	**0.7227**	**0.5939**
	RiDe [59]	PCA	0.8993	0.8581	0.8210	0.6863	0.5600
	MiLaN [60, 61]	Hashing	0.8472	0.8059	0.7987	0.6745	0.4205
	T-NLNN [62]	PCA	0.8670	0.8248	0.7710	0.6051	0.4900
	JSST (ours)	FC	0.9089	0.8729	0.8333	0.6874	0.5634
	JSST (ours)	PCA	**0.9290**	**0.8871**	**0.8536**	0.7118	0.5900
RSSCN	ResNet34 (baseline)	FC	0.8297	0.7754	0.7562	0.6755	0.6266
	ResNet34	PCA	0.8562	0.8129	0.7884	0.6862	0.6188
	SE [25]	PCA	0.8757	0.8407	0.8187	0.7423	0.6918
	SK [26]	PCA	0.8636	0.8229	0.7991	0.7214	0.6727
	NTS-Net [27]	PCA	0.8437	0.7914	0.7686	0.6808	0.6281
	DFLA [22]	PCA	0.7071	0.6311	0.6141	0.5512	0.5173
	LDCNN [20]	Max Pooling	0.6805	0.5954	0.5812	0.531	0.4988
	DHN [56]	Hashing	0.8351	0.6721	0.6021	0.4906	0.4521
	CBP [23]	PCA	0.8132	0.7550	0.7312	0.6542	0.6045
	RiDe [59]	PCA	0.8051	0.7443	0.7216	0.6383	0.5897
	MiLaN [60, 61]	Hashing	0.7964	0.7954	0.7560	0.5614	0.4525
	T-NLNN [62]	PCA	0.8281	0.7732	0.7489	0.6723	0.6161
	JSST (ours)	FC	0.8406	0.7907	0.7696	0.6964	0.6488
	JSST (ours)	PCA	**0.8889**	**0.8475**	**0.8307**	**0.7658**	**0.7156**

TABLE II
RETRIEVAL RESULTS OF DIFFERENT METHODS ON THE EXTRAORDINARY LARGE AID AND NWPU DATASETS

AID	ResNet34 (baseline)	FC	0.7687	0.6993	0.6680	0.5695	0.5118
	ResNet34	PCA	0.7925	0.7275	0.6940	0.5831	0.5124
	SE [25]	PCA	0.8237	0.7785	0.7489	0.6534	0.5973
	SK [26]	PCA	0.7959	0.7399	0.7108	0.6194	0.5661
	NTS-Net [27]	PCA	0.6795	0.5821	0.5439	0.4305	0.3774
	JSST (ours)	FC	0.7860	0.7292	0.6986	0.6065	0.5503
	JSST (ours)	PCA	**0.8286**	**0.7844**	**0.7583**	**0.6746**	**0.6205**
NWPU	ResNet34 (baseline)	FC	0.7389	0.6657	0.6362	0.5486	0.5039
	ResNet34	PCA	0.7686	0.7017	0.6698	0.5789	0.5268
	SE [25]	PCA	0.7931	0.7301	0.7006	0.6152	0.5698
	SK [26]	PCA	0.7520	0.6858	0.6567	0.5777	0.5348
	NTS-Net [27]	PCA	0.6092	0.5079	0.4755	0.3839	0.3392
	JSST (ours)	FC	0.7533	0.6819	0.6523	0.5699	0.5249
	JSST (ours)	PCA	**0.8039**	**0.7442**	**0.7156**	**0.6348**	**0.5910**

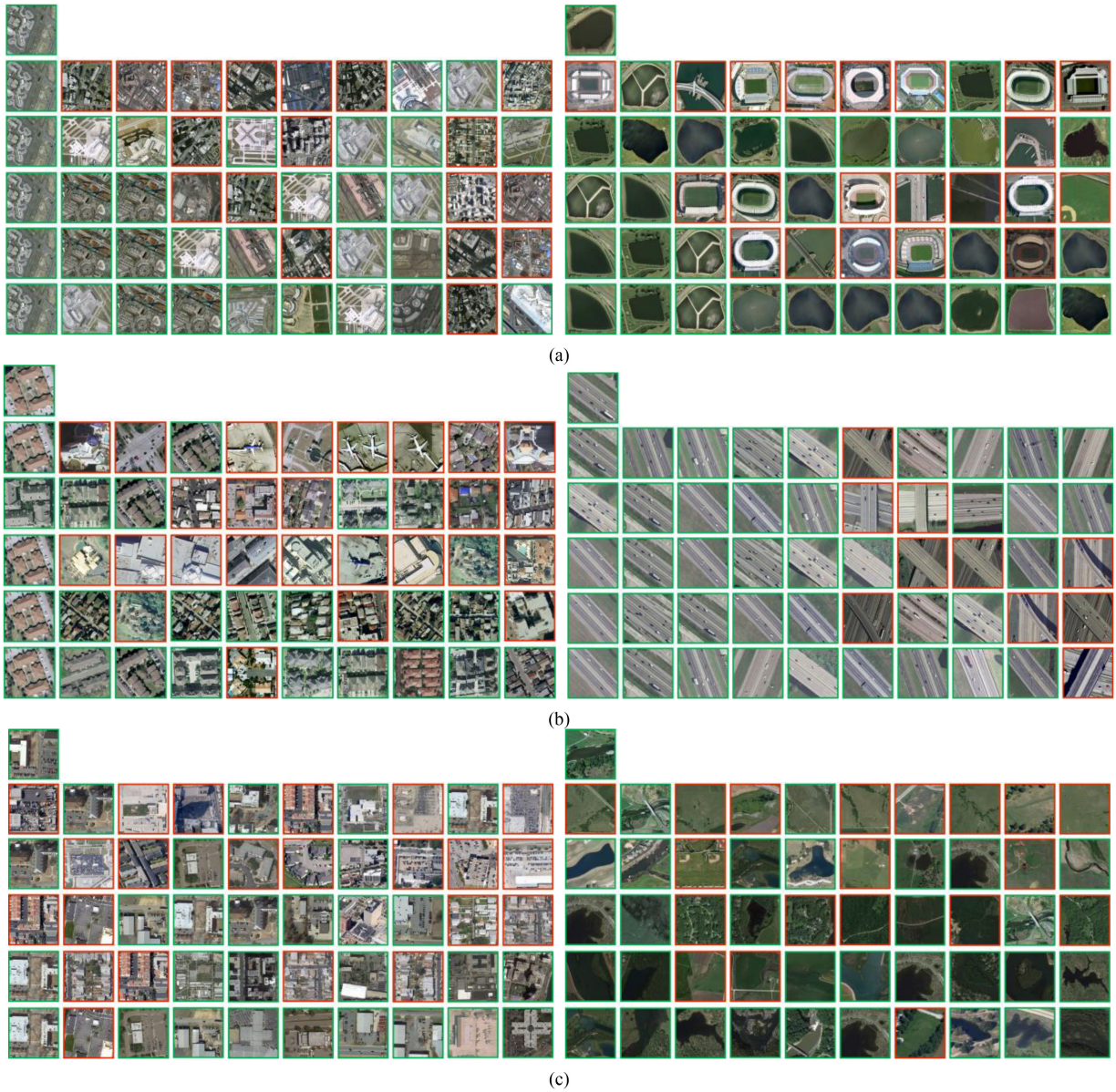


Fig. 3. Examples of retrieval results on (a) RS19, (b) UCM, and (c) RSSCN. The query images are shown in the first row, followed by the results of ResNet34(PCA), CBP, SENet, SKNet, and our JSST. The retrieval images are listed sequentially with their ranks.

The above examples demonstrate the difficulties of RSIR; and the subtle differences between related categories that challenge not only an algorithm but even humans. Even the well-known classification network ResNet was not able to fulfill this task entirely, which points up the difference between a classification problem and a retrieval problem. In any of our challenging cases, our JSST, which is based on integrated and learning-based spatial and spectral transformation, demonstrated its powerful discrimination ability.

2) Category Level Evaluation: We also compared the performance of our method to that of other methods at the category level. Fig. 4 visualizes the retrieval mAP of each category on the three datasets. The green, turquoise, blue, magenta, and red lines represent the methods of ResNet34, CBP, SENet, SKNet, and JSST with PCA reduction. Our JSST comprehensively outperformed the other methods with the exception of a few categories (e.g., resident and runway) where the performance of SENet exceeded JSST.

3) Effects of JSST: The greatest difference between JSST and the other methods is the novel learning-based transformation we imbedded in a deep learning model that aims to generate a new input image for better learning outcomes instead of the common revisions of building blocks in a CNN. The effects of JSST are visually demonstrated in Fig. 5, where, for each pair, the left image is the original input and the right image is the output of the spatial or spectral transformer. The cases in the first and second row clearly demonstrate the function of the spatial transformation, which predicts the foreground (region of interest) of an image and resamples it for highlighting. By doing this, a deep learning model can concentrate on the most informative region of an image from the interference of various and noisy backgrounds. The third row shows the effect

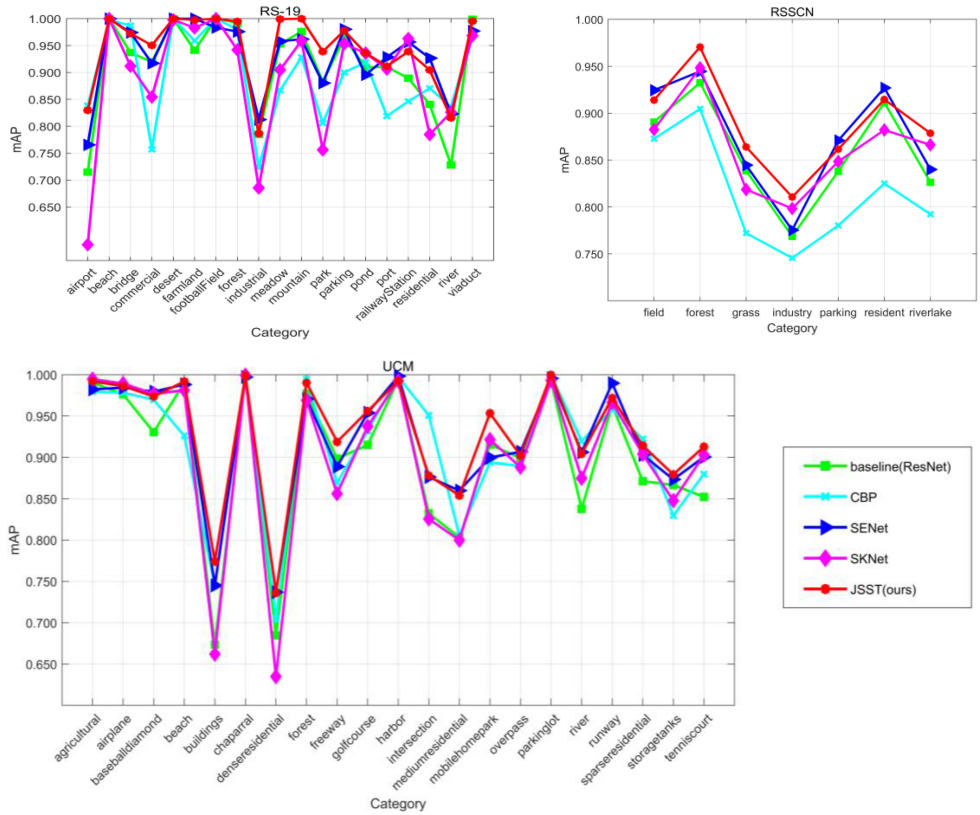


Fig. 4. Category-level mAP of different methods on the three datasets.

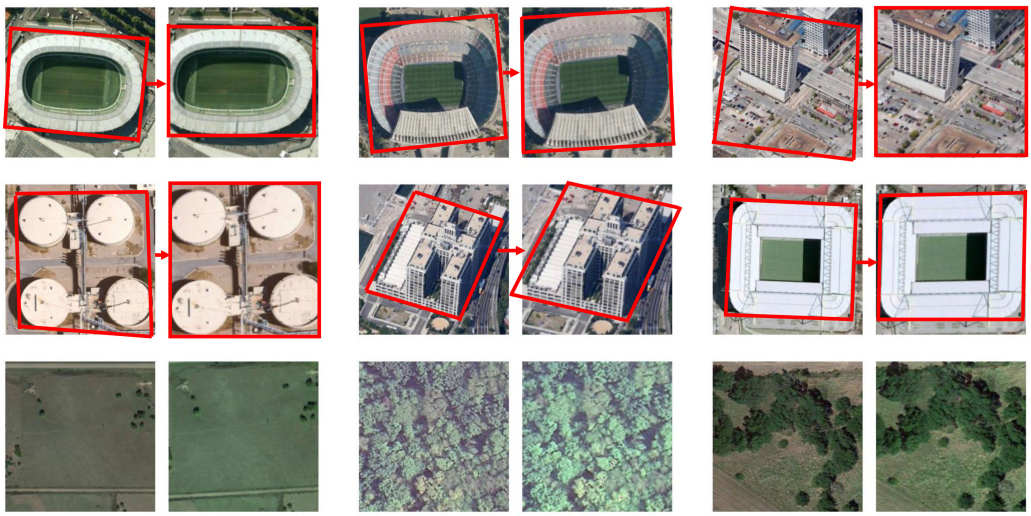


Fig. 5. Input images processed by JSST. The first and second rows are processed by the spatial transformation and the last row is processed by the spectral transformation.

of the spectral transformation, which changes the spectral bands of an input image to a better visual display, from which the deep learning model becomes more robust to changes in the radiometric conditions.

4) Effects of Parts: Our JSST is the combination of the spatial module and the spectral module. We tested the effectiveness of each module, and listed the results in Table III. Table III demonstrates that spatial or spectral transformation alone does improve the performance of the baseline, however, either of them is slightly worse than SE [25]. Their combination, i.e., JSST, has achieved the best.

D. Variations

We conducted several experiments regarding the alternative implementation of the proposed spatial and spectral transformation.

TABLE III
EXPERIMENTS WITH ONLY SPATIAL TRANSFORMATION OR ONLY SPECTRAL TRANSFORMATION. "SPATT" MEANS ONLY SPATIAL TRANSFORMATION USED. "SPECT" MEANS ONLY SPECTRAL TRANSFORMATION USED

Dataset	Method	mAP	P@5	P@10	P@50	P@100
RS19	ResNet34(baseline)	0.8867	0.8469	0.8087	0.5755	0.3854
	ResNet34(PCA)	0.9131	0.8745	0.8291	0.5474	0.3475
	SE [25]	0.9318	0.9051	0.8816	0.6593	0.4209
	SpatT	0.9250	0.8990	0.8740	0.6794	0.4264
	SpecT	0.9213	0.8980	0.8663	0.6656	0.4277
	JSST (ours)	**0.9432**	**0.9194**	**0.8964**	**0.6974**	**0.4354**
UCM	ResNet34(baseline)	0.9096	0.8605	0.8162	0.6669	0.5436
	ResNet34(PCA)	0.8994	0.8562	0.8040	0.6187	0.4758
	SE [25]	0.9167	0.8786	0.8419	0.7003	0.5771
	SpatT	0.9102	0.8776	0.8433	0.7063	0.5870
	SpecT	0.9170	0.8848	0.8474	0.7075	0.5891
	JSST (ours)	**0.9290**	**0.8871**	**0.8536**	**0.7118**	**0.5900**
RSSCN	ResNet34(baseline)	0.8297	0.7754	0.7562	0.6755	0.6266
	ResNet34(PCA)	0.8562	0.8129	0.7884	0.6862	0.6188
	SE [25]	0.8757	0.8407	0.8187	0.7423	0.6918
	SpatT	0.8736	0.8325	0.8111	0.7307	0.6720
	SpecT	0.8666	0.8268	0.8043	0.7188	0.6632
	JSST (ours)	**0.8889**	**0.8475**	**0.8307**	**0.7658**	**0.7156**
AID	ResNet34(baseline)	0.7687	0.6993	0.6680	0.5695	0.5118
	ResNet34(PCA)	0.7925	0.7275	0.6940	0.5831	0.5124
	SE [25]	0.8237	0.7785	0.7489	0.6534	0.5973
	SpatT	0.8131	0.7652	0.743	0.6592	0.6076
	SpecT	0.8131	0.7703	0.7410	0.6551	0.6018
	JSST (ours)	**0.8286**	**0.7844**	**0.7583**	**0.6746**	**0.6205**
NWPU	ResNet34(baseline)	0.7389	0.6657	0.6362	0.5486	0.5039
	ResNet34(PCA)	0.7686	0.7017	0.6698	0.5789	0.5268
	SE [25]	0.7931	0.7301	0.7006	0.6152	0.5698
	SpatT	0.7727	0.7092	0.6805	0.6014	0.5584
	SpecT	0.7698	0.7060	0.6789	0.5987	0.5567
	JSST (ours)	**0.8039**	**0.7442**	**0.7156**	**0.6348**	**0.5910**

TABLE IV
DIFFERENT SPATIAL TRANSFORMATION (TPS AND AFFINE) IMBEDDED IN OUR JSST ON THE THREE DATASETS

Dataset	Method	mAP	P@5	P@10	P@50	P@100
RS19	ResNet34(baseline)	0.8867	0.8469	0.8087	0.5755	0.3854
	ResNet34(PCA)	0.9131	0.8745	0.8291	0.5474	0.3475
	JSST with TPS	0.9312	**0.9112**	0.8765	0.6778	**0.4326**
	JSST with Affine	**0.9323**	0.9092	**0.8827**	**0.6889**	0.4316
UCM	ResNet34(baseline)	0.9096	0.8605	0.8162	0.6669	0.5436
	ResNet34(PCA)	0.8994	0.8562	0.8040	0.6187	0.4758
	JSST with TPS	0.9201	**0.8929**	0.8555	**0.7157**	**0.5926**
	JSST with Affine	**0.9233**	0.8919	**0.8579**	0.7116	0.5861
RSSCN	ResNet34(baseline)	0.8297	0.7754	0.7562	0.6755	0.6266
	ResNet34(PCA)	0.8562	0.8129	0.7884	0.6862	0.6188
	JSST with TPS	0.8682	0.8339	0.8098	0.7235	0.6724
	JSST with Affine	**0.8707**	**0.8368**	**0.8155**	**0.7421**	**0.6904**

1) Spatial Transformation: Two classic geometric transformations, TPS [46] and affine transformation, are integrated in JSST. The length and width of the TPS sampling grid were set to 6. The retrieval results of different spatial transformations on the three datasets are given in Table IV. It can be seen that adding a spatial transformation module, either TPS or affine, significantly improved the retrieval accuracy, and the affine transformation performed slightly better. TPS implements nonrigid transformation by fitting the control points with the minimum curvature surface, which may not be as suitable as affine transformation for rigid objects/scenes in a remote sensing image.

2) Spectral Transformation: As for the spectral transformation, we conducted experiments from two points of view (i.e., intrachannel and interchannel). The intrachannel spectral conversion experiment performed gamma transformations [41] on pixels of different positions to handle the local radiometric distortions of an image patch. In the interchannel spectral conversion experiment, we applied linear transformation with independent parameters to the three channels (i.e., the pixel-level conversion within one channel is uniform). All the transformation parameters of the two types were generated from PGN. The retrieval results of the different spectral transformations on the three datasets are given in Table V. It can be seen that the introduction of intra-channel conversion did not boost the retrieval performance, whereas the interchannel conversion was shown effective on all three datasets, with the latter adjusting the relative brightness difference between the RGB channels to simulate most of the overall illumination change, color bias,

TABLE V
COMPARISON OF DIFFERENT SPECTRAL TRANSFORMATION ON THREE REMOTE SENSING RETRIEVAL DATASETS. INTRA-C IS INTRACHANNEL TRANSFORMATION (GAMMA TRANSFORMATION); INTER-C IS INTERCHANNEL TRANSFORMATION (LINEAR TRANSFORMATION)

Dataset	Method	mAP	P@5	P@10	P@50	P@100
RS19	ResNet34(baseline)	0.8867	0.8469	0.8087	0.5755	0.3854
	ResNet34(PCA)	0.9131	0.8745	0.8291	0.5474	0.3475
	JSST with Intra-C	0.9003	0.8724	0.8357	0.6336	0.4064
	JSST with Inter-C	**0.9198**	**0.8908**	**0.8658**	**0.6578**	**0.4223**
UCM	ResNet34(baseline)	0.9096	0.8605	0.8162	0.6669	0.5436
	ResNet34(PCA)	0.8994	0.8562	0.8040	0.6187	0.4758
	JSST with Intra-C	0.9012	0.8538	0.8212	0.6629	0.5411
	JSST with Inter-C	**0.9144**	**0.8810**	**0.8476**	**0.7012**	**0.5688**
RSSCN	ResNet34(baseline)	0.8297	0.7754	0.7562	0.6755	0.6266
	ResNet34(PCA)	0.8562	0.8129	0.7884	0.6862	0.6188
	JSST with Intra-C	0.8536	0.8104	0.7823	0.6955	0.6398
	JSST with Inter-C	**0.8660**	**0.8211**	**0.7986**	**0.7105**	**0.6569**

TABLE VI
DIFFERENT COMBINATION MANNERS OF SPATIAL AND SPECTRAL TRANSFORMATION ON THREE REMOTE SENSING RETRIEVAL DATASETS. SPEC+SPAT MEANS SPECTRAL TRANSFORMATION IS EXECUTED BEFORE SPATIAL TRANSFORMATION

Dataset	Method	mAP	P@5	P@10	P@50	P@100
RS19	ResNet34(baseline)	0.8867	0.8469	0.8087	0.5755	0.3854
	ResNet34(PCA)	0.9131	0.8745	0.8291	0.5474	0.3475
	Hybrid	0.9156	0.8898	0.8546	0.6640	0.4230
	Spec + Spat	0.9052	0.8765	0.8556	0.6704	0.4260
	Spat + Spec	**0.9432**	**0.9194**	**0.8964**	**0.6974**	**0.4354**
UCM	ResNet34(baseline)	0.9096	0.8605	0.8162	0.6669	0.5436
	ResNet34(PCA)	0.8994	0.8562	0.8040	0.6187	0.4758
	Hybrid	0.9228	**0.8924**	0.8486	**0.7130**	**0.5919**
	Spec + Spat	0.9106	0.8757	0.8402	0.7010	0.5850
	Spat + Spec	**0.9290**	0.8871	**0.8536**	0.7118	0.5900
RSSCN	ResNet34(baseline)	0.8297	0.7754	0.7562	0.6755	0.6266
	ResNet34(PCA)	0.8562	0.8129	0.7884	0.6862	0.6188
	Hybrid	0.8659	0.8271	0.8039	0.7191	0.6655
	Spec + Spat	0.8702	0.8200	0.7957	0.7140	0.6640
	Spat + Spec	**0.8889**	**0.8475**	**0.8307**	**0.7658**	**0.7156**

and atmosphere conditions. This experiment verified that local spectral distortion is not important; and as we indicated before, we did image retrieval only on the cropped small patches of original images. Treating the spectral distortion channel-wise is a better choice.

These two experiments proved that either the spatial transformation or the spectral transformation on their own can effectively improve the accuracy of retrieval task. The manner of combining them was further investigated.

3) Arrangement of the Spatial and Spectral Transformation: Three strategies were tested: placing the spatial transformation at the front of the spectral transformation; placing the spectral transformation first; and implementing a hybrid mode. The hybrid mode consisted of performing the affine transformation of the spatial transformation and the spectral transformation on the original input images, but the interpolation of the spatial transformation is performed on the output images of spectral conversion to generate the output of the JSST. The results of the different strategies on three datasets are given in Table VI.

It can be seen that the sequential spatial-spectral mode performed better for the RS retrieval task, although the hybrid mode also improved the performance. Putting the spatial transformer before the spectral transformer achieved the best performance because the spatial transformation emphasizes the foreground objects/scenes, which further boosts the spectral transformation, whereas the facilitating role of the latter to the former is not that explicit.

4) Parameter Generation Network: PGN can be executed in two schemes: learning the parameters of spatial and spectral transformations independently or simultaneously. The former refers to the PGN having two separate feature extraction branches and outputs of six spatial and four spectral parameters independently. The latter refers to PGN performing feature extraction on the input image only once to output the whole 10 conversion parameters, which was our approach.

The retrieval results of the two strategies of PGN on three datasets are given in Table VII, where "PGN-I" represents the independence strategy, and "PGN-S" represents the simultaneity strategy.

Table VII gives that both of the strategies can effectively improve the effect of image retrieval and that the simultaneity scheme is superior. These results also indicate that the training of two branches may be more difficult than a branch, considering that the two branches outputting the same type of parameters (float scalar) can theoretically merge.

IV. DISCUSSION

The effectiveness and strength of our JSST method lies in the adjustment of sample space. The direct adjustment of sample

TABLE VII
RETRIEVAL RESULTS FOR THE TWO STRATEGIES OF PGN ON THE THREE DATASETS. PGN-I IS THE INDEPENDENCE STRATEGY. PGN-S IS SIMULTANEITY STRATEGY

Dataset	Architecture	mAP	P@5	P@10	P@50	P@100
RS19	ResNet34(baseline)	0.8867	0.8469	0.8087	0.5755	0.3854
	ResNet34(PCA)	0.9131	0.8745	0.8291	0.5474	0.3475
	PGN-I	0.9285	0.9051	0.8781	0.6673	0.4240
	PGN-S	**0.9432**	**0.9194**	**0.8964**	**0.6974**	**0.4354**
UCM	ResNet34(baseline)	0.9096	0.8605	0.8162	0.6669	0.5436
	ResNet34(PCA)	0.8994	0.8562	0.8040	0.6187	0.4758
	PGN-I	0.9196	0.8829	0.8469	0.7039	0.5816
	PGN-S	**0.9290**	**0.8871**	**0.8536**	**0.7118**	**0.5900**
RSSCN	ResNet34(baseline)	0.8297	0.7754	0.7562	0.6755	0.6266
	ResNet34(PCA)	0.8562	0.8129	0.7884	0.6862	0.6188
	PGN-I	0.8579	0.8068	0.7886	0.7160	0.6627
	PGN-S	**0.8889**	**0.8475**	**0.8307**	**0.7658**	**0.7156**

TABLE VIII
RETRIEVAL RESULTS OF DIFFERENT SAMPLE PROCESSING METHODS. THE BASELINE IS RESNET34 WITH PCA. ITERATIONS 12 AND 3 OF SELF-TRAINING HAVE TRAINING SAMPLES OF 2100 + 6057, 2100 + 14722, AND 2100 + 21344, RESPECTIVELY. THE SMALL UCM IS USED AS THE TRAINING SET. THE SELF-TRAINING INTRODUCES NEW SAMPLES FROM PATTERNNET (TREATED AS UNLABELED)

Dataset	Method		mAP	P@5	P@10	P@50	P@100
RS19	ResNet34 (baseline)		0.8034	0.7500	0.7015	0.4955	0.3439
	Self-training [64]	Iteration 1	0.8279	0.7765	0.7459	0.5314	0.3705
		Iteration 2	0.8306	0.7776	0.7434	0.5427	0.3785
		Iteration 3	0.8511	0.8031	0.7679	**0.5609**	**0.3863**
	JSST+ResNet34 (ours)		**0.8600**	**0.8112**	**0.7709**	0.5392	0.3768
RSSCN	ResNet34 (baseline)		0.7746	0.7107	0.6900	0.6229	0.5828
	Self-training [64]	Iteration 1	0.7854	0.7257	0.7052	0.6385	0.6011
		Iteration 2	0.7962	0.7407	0.7205	0.6530	0.6097
		Iteration 3	0.7926	0.7350	0.7171	0.6550	0.6186
	JSST+ResNet34 (ours)		**0.8046**	**0.7511**	**0.7293**	**0.6770**	**0.6428**

space is apparently more efficient compared to data augmentation as the latter expands sample space through resampling the original labeled inputs to many counterparts. Another method for expanding sample space is self-training, which incorporates the unlabeled samples as the labeled ones gradually. We conducted experiments to compare the performance of our JSST and the very recent self-training method [64], which uses a trained model from a small labeled dataset to label unlabeled data and then iteratively updates the training dataset by adding those highly-reliable samples for the unlabeled dataset to train a better model. We chose UCM [52] as the labeled small dataset with 2100 labeled samples, and PatternNet [50] as the supplementary unlabeled dataset. There were three iterations of self-training; and for each iteration, the number of training samples was 2100 + 6057 (iteration 1), 2100 + 14722 (iteration 2), and 2100 + 21344 (iteration 3). The labeling confidence was set to 0.85 (i.e., an unlabeled sample was added to the training dataset when its confidence exceeded 0.85). The retrieved datasets were RS-19 [51] and RSSCN [53]. The results are given in Table VIII where it can be seen that JSST, with much less training samples, achieved higher accuracy than [64] on the retrieval datasets. From the limited experiments, it can be seen that the adjustment of the sample space was more elegant and accurate than the expansion of the sample space for image retrieval.

SGD and Adam are two of the mainstream gradient descent methods. We found that using SGD instead of Adam [65] allowed not only faster convergence in the training phase, but also achieved higher accuracy in the retrieval phase in most of the experiments. The only exception was CBP [23], where SGD caused training divergence, which may have been caused by the backpropagation of special bilinear pooling. Note that although in-depth review of the theoretical difference between Adam and SGD was beyond the scope of this article, a review of the past literature is worthwhile here. For example, Wilson *et al*. [66] believed that the use of an adaptive learning rate method such as Adam may overemphasize the features that appear in the early stage, and as a result, may suppress the contribution of the features in the later stage. Reddi *et al*. [67] experimentally proved that Adam did not converge in some cases. Keskar and Socher [68] conducted a comparative experiment on the CIFAR-10 dataset and found that although Adam had a faster convergence speed, the final effect was not as good as that of the group using SGD.

We used ResNet34 as our lightweight backbone and did not consider heavier backbones (e.g., ResNet101 or Hourglass104 [69]) for the sake of efficiency. The other commonly used lightweight backbone is VGG [70]. However, the number of parameters and amount of calculation of ResNet34 are only a fifth and a fourth of that of VGG16. Furthermore, we have experimentally demonstrated that ResNet performed better than VGG for RSIR in our previous work [23].

In this article, our JSST is applied to high resolution or very high resolution RSIR. However, we believe that it can also benefit the multispectral or hyperspectral image retrieval. However, there is few such dataset (e.g., EuroSAT Dataset [71]) for scene classification or image retrieval up to now. If more multispectral or hyperspectral image retrieval datasets are available, the only modification of the proposed JSST is to set

accordingly the output vector length of the regression layer in the spectral module.

Besides its superiority when compared to other recent methods in our extensive experiments, JSST also has the advantage of being a generic and plug-in front-end module of a deep learning-based image retrieval network that can be implemented in most of the other retrieval methods, especially those emphasizing a network structure or back-end pooling/encoding improvement but lacking the pre-processing of inputs, for example, JSST can provide a lot of room for boosting the performance of advanced retrieval methods such as SENet [25] and CBP [23].

V. Conclusion

In this article, we proposed a lightweight module to perform specific geometric and spectral modification on the input image called the JSST model. Our JSST automatically learns the spatial and spectral parameters through a PGN to convert the original input image into an output image, which can then be better described and discriminated by the image retrieval network through emphasizing the objects/scenes of interest and correcting spectral distortions.

We conducted extensive experiments on four RSIR datasets. Our results demonstrated that introducing the JSST made a significant difference when compared with a baseline method and state-of-the-art models.

References

[1] H. Lotz-Iwen and W. Steinborn, "The intelligent satellite-image information system ISIS," in *Proc. AIP Conf.*, 1993, pp. 727–734.

[2] C. Chang, B. Moon, A. Acharya, C. Shock, A. Sussman, and J. Saltz, "Titan: A high-performance remote-sensing database," in *Proc. 13th Int. Conf. Data Eng.*, Apr. 1997, pp. 375–384.

[3] G. B. Marchisio, W.-H. Li, M. Sannella, and J. R. Goldschneider, "GeoBrowse: An integrated environment for satellite image retrieval and mining," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 1998, vol. 2, pp. 669–673.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[5] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[6] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 886–893.

[7] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2003, pp. 1470–1477.

[8] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 3304–3311.

[9] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[10] L. Bo and C. Sminchisescu, "Efficient match kernel between sets of features for visual recognition," in *Proc. Neural Inf. Process. Syst.*, Dec. 2009, pp. 135–143.

[11] A. Iscen, T. Furon, V. Gripon, M. Rabbat, and H. Jégou, "Memory vectors for similarity search in high-dimensional spaces," *IEEE Trans. Big Data,* vol. 4, no. 1, pp. 65–77, Mar. 2017.

[12] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.,* vol. 61, pp. 85–117, Jan. 2015.

[13] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found. Trends Signal Process.,* vol. 7, no. 3/4, pp. 197–387, Jun. 2014.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 2, pp. 1097–1105, 2012.

[15] A. Mousavian and J. Kosecka, "Deep convolutional features for image based retrieval and scene categorization," 2015, *arXiv:1509.06033*, [Online]. Available: https://arxiv.org/abs/1509.06033

[16] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2014, pp. 392–407.

[17] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," 2013, *arXiv:1312.6229*, [Online]. Available: https://arxiv.org/abs/1312.6229

[18] 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.

[19] B. Demir and L. Bruzzone, "Hashing-based scalable remote sensing image search and retrieval in large archives," *IEEE Trans. Geosci. Remote Sens.,* vol. 54, no. 2, pp. 892–904, Sep. 2016.

[20] W. Zhou, S. Newsam, C. Li, and Z. Shao, "Learning low dimensional convolutional neural networks for high-resolution remote sensing image retrieval," *Remote Sens.,* vol. 9, no. 5, pp. 489–508, May 2017.

[21] X. Tang, L. Jiao, and W. J. Emery, "SAR image content retrieval based on fuzzy similarity and relevance feedback," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.,* vol. 10, no. 5, pp. 1824–1842, May 2017.

[22] W. Xiong, Y. Lv, Y. Cui, X. Zhang, and X. Gu, "A discriminative feature learning approach for remote sensing image retrieval," *Remote Sens.,* vol. 11, no. 3, pp. 281, Feb. 2019.

[23] Y. Wang, S. Ji, M. Lu, and Y. Zhang, "Attention boosted bilinear pooling for remote sensing image retrieval," *Int. J. Remote Sens.,* vol. 41, no. 7, pp. 2704–2724, Dec. 2019.

[24] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3456–3465.

[25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

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

[27] Z. Yang, T. Luo, D. Wang, Z. Hu, J. Gao, and L. Wang, "Learning to navigate for fine-grained classification," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 420–435.

[28] Y. Liu, L. Ding, C. Chen, and Y.-B. Liu, "Similarity-based unsupervised deep transfer learning for remote sensing image retrieval," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 11, pp. 7872–7889, Apr. 2020.

[29] A. B. Yandex and V. Lempitsky, "Aggregating deep convolutional features for image retrieval," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1269–1277.

[30] Y. Kalantidis, C. Mellina, and S. Osindero, "Cross-dimensional weighting for aggregated deep convolutional features," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 685–701.

[31] J. Yue-Hei Ng, F. Yang, and L. S. Davis, "Exploiting local features from deep networks for image retrieval," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit., Deep Vis. Workshop*, Jun. 2015, pp. 53–61.

[32] A. Radoi and M. Datcu, "Multilabel annotation of multispectral remote sensing images using error-correcting output codes and most ambiguous examples," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 7, pp. 2121–2134, Jul 2019.

[33] A. Kosiorek, S. Sabour, Y. W. Teh, and G. E. Hinton, "Stacked capsule autoencoders," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2019, pp. 15512–15522.

[34] J. Dai *et al.*, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 764–773.

[35] C. Esteves, C. Allen-Blanchette, X. Zhou, and K. Daniilidis, "Polar transformer networks," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–14.

[36] M. Jaderberg, K. Simonyan, and A. Zisserman, "Spatial transformer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2015, pp. 2017–2025.

[37] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3642–3649.

[38] A. Kanazawa, A. Sharma, and D. Jacobs, "Locally scale-invariant convolutional neural networks," 2014, *arXiv:1412.5104*, [Online]. Available: https://arxiv.org/abs/1412.5104

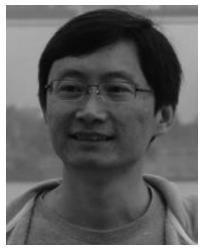
[39] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys, "TI-POOLING: Transformation-invariant pooling for feature learning in convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 289–297.

[40] J. A. Stark, "Adaptive image contrast enhancement using generalizations of histogram equalization," *IEEE Trans. Image Process.*, vol. 9, no. 5, pp. 889–896, May 2000.

[41] S.-C. Huang, F.-C. Cheng, and Y.-S. Chiu, "Efficient contrast enhancement using adaptive gamma correction with weighting distribution," *IEEE Trans. Image Process.*, vol. 22, no. 3, pp. 1032–1041, Mar. 2013.
[42] D. Li, M. Wang, and J. Pan, "Auto-dodging processing and its application for optical RS images," *Geomatics Inf. Sci. Wuhan Univ.,* vol. 31, no. 9, pp. 753–756, Sep. 2006.
[43] J.-M. Morel and G. Yu, "ASIFT: A new framework for fully affine invariant image comparison," *SIAM J. Imag. Sci.*, vol. 2, no. 2, pp. 438–469, Apr. 2009.
[44] K. Zhang, N. Snavely, and J. Sun, "Leveraging vision reconstruction pipelines for satellite imagery," in *Proc. IEEE Int. Conf. Comput. Vis. workshop*, 2019, *arXiv:1910.02989*, [Online]. Available: https://arxiv.org/abs/1910.02989.
[45] K. T. Gribbon and D. G. Bailey, "A novel approach to real-time bilinear interpolation," in *Proc. 2nd IEEE Int. Workshop Electron. Des., Test Appl.*, 2004, pp. 126–131.
[46] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 6, pp. 567–585, Jun. 1989.
[47] M. A. Nuno-Maganda and M. O. Arias-Estrada, "Real-time FPGA-based architecture for bicubic interpolation: An application for digital image scaling," in *Proc. Int. Conf. Reconfigurable Comput. FPGAs*, Sep. 2005, pp. 8–11.
[48] 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.
[49] 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.*, 2009, pp. 248–255.
[50] W. Zhou, S. Newsam, C. Li, and Z. Shao, "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval," *ISPRS J. Photogram. Remote Sens.*, vol. 145, pp. 197–209, Nov. 2018.
[51] G.-S. Xia, W. Yang, J. Delon, Y. Gousseau, H. Sun, and H. Maître, "Structural high-resolution satellite image indexing," in *Proc. Symp., 100 Years ISPRS—Adv. Remote Sens. Sci.*, Jul. 2010.
[52] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proc. ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Nov. 2010, pp. 270–279.
[53] Q. Zou, L. Ni, T. Zhang, and Q. Wang, "Deep learning based feature selection for remote sensing scene classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 11, pp. 2321–2325, Nov. 2015.
[54] 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.
[55] 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.
[56] H. Zhu, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 2415–2421.
[57] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Representations*, 2013, *arXiv:1312.4400*, [Online]. Available: https://arxiv.org/abs/1312.4400
[58] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2010, pp. 143–156.
[59] J. Kang *et al.*, "Rotation-Invariant deep embedding for remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, to be published, doi: 10.1109/TGRS.2021.3088398.
[60] S. Roy, E. Sangineto, B. Demir, and S. Nicu,"Metric-learning-based deep hashing network for content-based retrieval of remote sensing images," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 2, pp. 226–230, Feb. 2021.
[61] S. Roy, E. Sangineto, B. Demir, and S. Nicu, " Deep metric and hash-code learning for content-based retrieval of remote sensing images," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2018, pp. 4539–4542.
[62] M. Zhang, Q. Cheng, F. Luo, and L. Ye, "A triplet nonlocal neural network with dual-anchor triplet loss for high-resolution remote sensing image retrieval". *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 2711–2723, Feb. 2021.
[63] X.-Y. Tong, G.-S. Xia, F. Hu, Y. Zhong, M. Datcu, and L. Zhang, "Exploiting deep features for remote sensing image retrieval: A systematic investigation," *IEEE Trans. Big Data*, vol. 6, no. 3, pp. 507–521, Oct. 2019.
[64] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10687–10698.
[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–15.
[66] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2017, pp. 4148–4158.
[67] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam and beyond," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–23.
[68] N. S. Keskar and R. Socher, "Improving generalization performance by switching from ADAM to SGD," 2017, *arXiv:1712.07628*, [Online]. Available: https://arxiv.org/abs/1712.07628
[69] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 483–499.
[70] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–13.
[71] 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 Sens.*, vol. 12, no. 7, pp. 2217–2226, Jul. 2019.

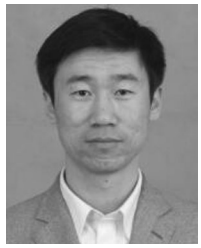
Yameng Wang received the B.S. degree in remote sensing science and technology in 2018 from Wuhan University, Wuhan, China, where she is currently working toward the M.S. degree in photogrammetry and remote sensing with the School of Remote Sensing and Information Engineering.

Her research interests include remote sensing image processing and machine learning.



Shunping Ji (Member, IEEE) received the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan, China, in 2007.

He is currently a Professor with the School of Remote Sensing and Information Engineering, Wuhan University. He has coauthored more than 60 articles. His research interests include photogrammetry, remote sensing image processing, mobile mapping system, and machine learning.



Yongjun Zhang received the B.S., M.S., and Ph.D. degrees in photogrammetry and remote sensing from Wuhan University (WHU), Wuhan, China, in 1997, 2000, and 2002, respectively.

He is currently a Professor of photogrammetry and remote sensing with the School of Remote Sensing and Information Engineering, WHU. His research interests include aerospace and low-attitude photogrammetry, image matching, combined block adjustment with multisource data sets, integration of LiDAR point clouds and images, and three-dimensional city reconstruction.