Invariant Deep Compressible Covariance Pooling for Aerial Scene Categorization

Shidong Wang, Yi Ren, Gerard Parr *Member, IEEE* Yu Guan and Ling Shao, *Senior Member, IEEE*

Abstract—Learning discriminative and invariant feature representation is the key to visual image categorization. In this article, we propose a novel invariant deep compressible covariance pooling (IDCCP) to solve nuisance variations in aerial scene categorization. We consider transforming the input image according to a finite transformation group that consists of multiple confounding orthogonal matrices, such as the D4 group. Then, we adopt a Siamese-style network to transfer the group structure to the representation space, where we can derive a trivial representation that is invariant under the group action. The linear classifier trained with trivial representation will also be possessed with invariance. To further improve the discriminative power of representation, we extend the representation to the tensor space while imposing orthogonal constraints on the transformation matrix to effectively reduce feature dimensions. We conduct extensive experiments on the publicly released aerial scene image data sets and demonstrate the superiority of this method compared with state-of-the-art methods. In particular, with using ResNet architecture, our IDCCP model can reduce the dimension of the tensor representation by about 98% without sacrificing accuracy (i.e., $<0.5\%$).

Index Terms—Invariant Feature Representation, Symmetric Positive Definite (SPD) Manifold, Stiefel Manifold and Aerial Scene Categorization.

I. INTRODUCTION

Aerial scene classification, also known as remote sensing scene classification, is considered to be one of the most active tasks in scene classification. The classification of aerial scene images involves a wide range of applications, such as environment monitoring, urban and agricultural planning, and land use and land cover (LULC) classification [1], [2], [3], [4]. The recent development of remote sensing technologies leads to the accumulation of very high spatial resolution images (e.g., ~1-4 m/pixel), which takes out aerial imagery characteristics to the new level of illustrating the geometry structure and texture peculiarities in a more distinct way. The increasingly spatial resolution of aerial images not only allows depicting image peculiarities across smaller spatial extents but also makes the classification more ambiguous and challenging.

Shidong Wang is with Open Lab, School of Computing, Newcastle University, Newcastle upon Tyne NE4 5TG, U.K., also with the School of Computing Sciences, University of East Anglia, Norwich NR4 7TJ, U.K. (e-mail: shidong.wang@ncl.ac.uk)
Yi Ren and Gerard Parr are with School of Computing Sciences, University of East Anglia, Norwich, U.K. (e-mail: e.ren and g.parr@uea.ac.uk)
Yu Guan is with Open Lab, School of Computing, Newcastle University, Newcastle upon Tyne NE4 5TG, U.K. (e-mail: yu.guan@ncl.ac.uk).
Ling Shao is with Inception Institute of Artificial Intelligence, Abu Dhabi, United Arab Emirates, e-mail:(ling.shao@ieee.org)
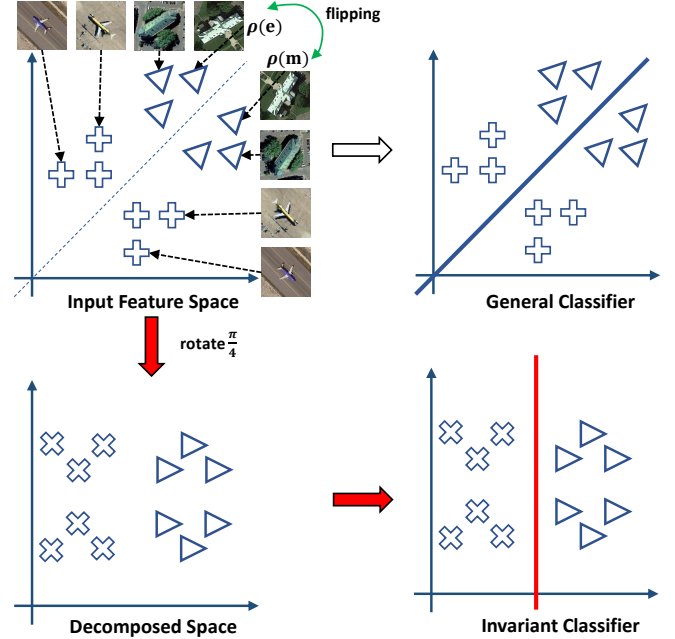

Fig. 1. Solely flipping the input image may make the conventional classifier inoperable. Combined with the rotation transformation, a new orthogonal representation space is formed. Then, we can infer a trivial representation from the space and leverage it to train our invariant classifier.

Compared with conventional scene images, the texture information of remote sensing images is more complicated. The main reason for sophisticated texture features is the variation of orientation, scale, and shape of objects in the image. In addition to these variations, the inherent property of remote sensing images is also quite different from the ordinary scene images. Precisely, remote sensing image, as one of the most representative overhead images, has no dominant left–right or up–down relationships. To classify a typical scene image, we only concern the absence or presence of the main object. However, in the aerial scene classification task, an expectation is that the model is capable of assigning the correct label for a given image regardless of its absolute orientation. This sought-after property remains strictly constant under all transformations of the input data, which is so-called *invariance*.

Invariance can be directly encoded and considered to be the most effective method to mitigate the impact of variations of the input data. However, incorporating invariant information is challenging, even for the powerful convolutional neural network (CNN) architectures. Precisely, off-the-shelf CNN architectures are only endowed with the minimal internal

structures due to the costly computing of the optimization. These minimal intrinsic structures are capable of handling locally minor shifts but not global transformations. Data augmentation [5] is widely adopted to incorporate the prior knowledge of input data, but there is no guarantee that the invariance learned in the training stage is effectively generalized for the test data. Furthermore, it is difficult to quantify the predominate transformations and lacks the interpretability of feature maps. In contrast to the redundant approaches, such as data augmentation, one of the latest research lines is toward procuring the equivariance from equivariant CNNs [6], [7], [8], [9]. The basic idea of these methods is to learn the *transformation-equivariant CNN* by constructing features in a linear *G*-space and then derive an invariant subspace by employing the appropriate pooling method (e.g., the coset pooling). These methods can detect co-occurrences of features at any *positions* in a standard CNN architecture, and any preferred *poses* in a *G*-space, but the computational cost scales dramatically with the increasing cardinality of the group.

To address the shortcomings of the aforementioned approaches, a novel framework is proposed to derive the transformation-invariant subspace from a finite linear *G*-group space, which allows group actions to be directly applied to the raw image. As shown in Fig. 1, merely flipping the local feature space can render the traditional classifier fail to work. Through looking insight into the flipping operation, we find that it can be expressed by the permutation matrices. The expression of permutation matrices implies two primary properties: the flipping operation acts orthogonally at the local pixel and prevents images from distortion during transformation. These motivate us to construct a transformation group *G* where all the decomposed spaces are orthogonal to each other (i.e., D4 group in our scenarios). An invariant feature space can be sought through using the reducible decomposition of the representations of *G*-space. Namely, it allows us to decompose the action of *G* into the direct sum of irreducible representations and results in a locally invariant subspace that serves to train an invariant classifier.

The orthogonal transformations prevent the pixel value shifting in the process of transforming but cannot avoid the changes of pixel locations. To alleviate the effect of pixel position changes, we can calculate the tensor product of irreducible representations to form a global representation based on the fact that the reducible decomposition of the representation conforms to the group action of *G*-space. The tensor representation contains more discriminative information than the conventional first-order feature but suffers from the high-dimensional problem. Considering that the second-order feature representation is a covariance matrix (i.e., symmetric positive definite (SPD) matrix), we can force the weight matrix to be a row full-rank matrix where all elements reside on a Stiefel manifold. In this way, it can produce a compact space while maintaining the geometry of the SPD manifold. Our contributions can be summarized as follows:

- We derive an invariant classifier from the learned weights of the trivial tensor representation with the guarantee of being invariant under the finite *G*-group actions.
- We introduce a way of imposing orthogonal constraints

on the weight matrix to effectively map the high-dimensional SPD manifolds into new compact manifolds.
- We conduct extensive experiments on four aerial scene image datasets and achieve state-of-the-art performance.

II. RELATED WORKS

A. Aerial Scene Classification

The last decade has witnessed a dramatic growth of research interests in aerial scene image classification. Early attempts heavily rely on the manually designed features. Globally handcrafted features summarize the overall statistical properties and can be directly fed into classifiers. Examples include color descriptors [10], [11], [12], texture descriptors [13], [14], [15] and histograms of oriented gradients (HOG) descriptors [16], [17]. Locally handcrafted features usually need to be transformed into the higher level representation using coding methods, such as the Bag-of-Visual-Words (BoVW) models [18], [19], [20] and the Fisher vector [21], [22], [15]. Shortly, unsupervised learning methods [23], [24], [25], [2], [26] become increasingly popular to remedy the limitations of handcrafted features. Typical unsupervised feature learning methods include, but not limited to, principal component analysis (PCA) [27], sparse coding [28], [29], [30], autoencoder [31], [32], [33], [34] and K-means clustering that often associates with BoVW methods. Benefiting from the capability of incorporating intricate structures hidden in high-dimensional data, deep learning-based methods present extensive popularity of being adopted for aerial image classification. For example, Cheng *et al.* [35] investigated the efficiency of the BoVW model utilizing convolutional features. [36] learned the rotation-invariant feature to improve the performance of object detection in remotely sensed images. Other works include transferring learning-based methods [37], [38], [39], [40], [41], [42], domain adaption [43] and feature fusion [44]. Feature fusion-based methods can also achieve gratifying results and have different forms that include multi-scale [45], multi-layer [46], multi-stream [44], [47] and multi-granularity [3]. For more details, we refer readers to [1] and [2], and a comprehensive review of remote sensing image interpretation based on deep learning techniques [4].

B. Equivariant/Invariant CNN

Existing methods capture the transformation-invariant information by transforming the inputs or the filters. For the former category, the most standard method is data augmentation [5]. This approach aims to increase the capacity of the model in terms of some specific variations by generating abundant training samples. RIFD-CNN [48] introduced an explicit regularization as a constraint that forces the model learning the invariance of CNN features. Ti-pooling [49] employed a parallel Siamese architecture to extract features from multiple rotated images and provided a pooling across the different features at the first fully-connected layer. Henriques and Vedaldi [9] proved that the inherent translation equivariance of CNNs could be worked on the warped images and achieved the transformation equivariance. Recently, Chen *et al.* [47] proposed a recurrent transformer network (RTN),

which exploited the spatial transformer network progressively and learned multiple transformation-invariant scales of the input image. For the latter category, existing methods usually construct features using group representations. For example, Cohen and Welling [6] learned feature invariance by using the coset pooling for an asymmetry transformation group space that is composed of dihedral flipping and four $90°$ rotations. This work had been extended in [7] to decouple the computational cost by performing the general, steerable representations. Instead of rotating filters, Dieleman *et al.* [8] proposed to rotate feature maps at four $90°$ rotations and preserved equivariance in CNNs. In addition, Mukuta and Harada [50] and Sokolic *et al.* [51] analyzed the significance of encoding first- or second-order invariant features in learning algorithms.

C. Second-order Statistics Feature

Bilinear pooling [52], as one of the most successful second-order pooling methods, collects second-order statistics of local CNN features over the entire image to form a holistic representation. DeepO$_2$P [53] performed a matrix back-propagation structure for both singular value decomposition (SVD) and eigenvalue decomposition (EIG). Improved bilinear pooling [54] investigated the performance of using the combination of different normalization methods, such as the matrix square root normalization, an element-wise square root, and l_2 normalisation. Acharya *et al.* [55] proposed a covariance pooling framework that exploited the Riemannian manifold for facial expression recognition. Various methods have been proposed to reduce the high-dimensional of the bilinear feature, for example, Random Maclaurin [56], Tensor Sketch [56], low-rank constraints [57], Gaussian RBF kernel [58] and Grassmann manifold [59]. Besides, iSQRT-COV [60] provided a method to speed up the calculating of the square root of the global covariance matrix by using Newton-Schulz iteration in both forward and backward propagations.

III. PRELIMINARY NOTIONS AND DEFINITIONS

We use calligraphic typeface $\mathcal{X}$ and $\mathcal{F}$ to denote the input image and the deep CNN features, respectively. A group $G = (\mathcal{X}, \bullet)$ is the pair of a set $\mathcal{X}$, together with an operation $\bullet : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{X}$ (also known as group law) that satisfies the group axioms of closure, associativity, identity and invertibility. The number of elements in a finite $\mathcal{X}$ is denoted as $|\mathcal{X}|$. A homomorphism is a map from a group G to the group of automorphisms of a vector space V that preserves group action operations, $\rho(g_1) \bullet \rho(g_2) = \rho(g_1 \bullet g_2), \forall g_1, g_2 \in G$ and exists the d-dimensional identity matrix $\rho(e) = 1_{d \times d}$. For a concrete example, $\rho : G \rightarrow GL(V)$ is a homomorphism and also called a representation, where GL is the general linear space. A representation is named a trivial representation if and only if it maps all $g \in G$ to $1_{d \times d}$ (e.g., one-dimensional trivial representation is denoted as **1**). Similarly, the representation is called a unitary representation or orthogonal representation when all $\rho(g)$ are unitary matrices or orthogonal matrices. The space of intertwining operator is written as $\text{Hom}_{\mathcal{X}}(\rho, \rho^{'})$ which implies that there is a linear operator $L : \mathbb{C}^d \rightarrow \mathbb{C}^d$

that satisfies $L \bullet \rho(g) = \rho^{'}(g) \bullet L$. If L is a bijective function that satisfies $L \in \text{Hom}_{\mathcal{X}}(\rho, \rho^{'})$, we will write it as $\rho \simeq \rho^{'}$. Given two representations (ρ, V_1) and (σ, V_2) of the same group G, the direct sum of these two representations is given as $\rho \oplus \sigma : G \rightarrow GL(V_1 \oplus V_2)$ with regarding G as block-diagonal form of $G \times G$. According to Schur's Lemma, $\text{Hom}_G(\rho_1, \rho_2) = \{0\}$ if ρ_1 and ρ_2 are not isomorphic or 1-D when they are isomorphic. If ρ and σ are in tensor spaces, the tensor representation will be denoted as $\rho \otimes \sigma$. The character function $\mathcal{X}_\rho$ that maps G into a finite-dimensional vector space over a filed F is given by $\mathcal{X}_\rho(g) = \text{tr}(\rho(g))$, where $\text{tr}(\cdot)$ is the trace operation. The degree of a representation ρ is the dimension of its representation space V and we denote it as $\dim(\rho)$.

IV. METHOD

A. Transformation-Equivariant Networks

In deep learning models, the transformation-equivariant preserves the capacity to capture various useful transformations. An example is the translation-equivariant in convolution layers, which can be exploited in any layers of the deep CNN architecture. Given an input image $\mathcal{X}$, the transformation equivariant can be regarded as seeking a unique $T_g^{'} \in G^{'}$ that satisfies:

$$\Phi(T_g(\mathcal{X})) = T_g^{'}(\Phi(\mathcal{X})) \tag{1}$$

where $T_g^{'}$ is an action in a group structure $G^{'}$ and Φ denotes the feature mapping function. For brevity, it is usually written as $\Phi(T_g(\mathcal{X})) = T_g(\Phi(\mathcal{X}))$ since $T_g^{'} = T_g$ and then $G^{'} = G$. However, we prefer the former format since $\Phi(\mathcal{X})$ and $T_g(\mathcal{X})$, perhaps, lie in the different domains. Two strategies can be derived from the definition to achieve the equivariance to transformations. On the one hand, $T_g^{'}(\Phi(\mathcal{X}))$ indicates an explicit way to learn equivariance of transformations by transforming kernels or feature maps extracted from the input image, such as [6], [8]. However, these methods are generally inefficient because they require complicated permutations of each convolution kernel in all convolutional layers and need retraining on large-scale datasets. In addition, they neglect the manipulation of shared weights between convolution kernels, which makes them difficult to transfer or scale to new challenging tasks. $\Phi(T_g(\mathcal{X}))$, on the other hand, offers an option to achieve transformation-equivariant by transforming input image directly. However, this branch arises less attention or has been referred to data augmentation method [5].

To cope with the abovementioned problems, we propose a novel framework to achieve equivariance by directly transforming input images and extracting the corresponding features with multiple CNNs. As shown in Fig 2, we first transform the input image according to a D_4 transformation group that consists of image reflections and rotations by multiples of $90°$. The main reason for choosing the D_4 group is that the group is a regular and symmetrical polygon. In other words, it implies that any actions in a D_4 group can prevent the image transformation from distortion. Once the transformed images have been obtained, we focus on seeking for an architecture that is effective to retain the group structure during the feature
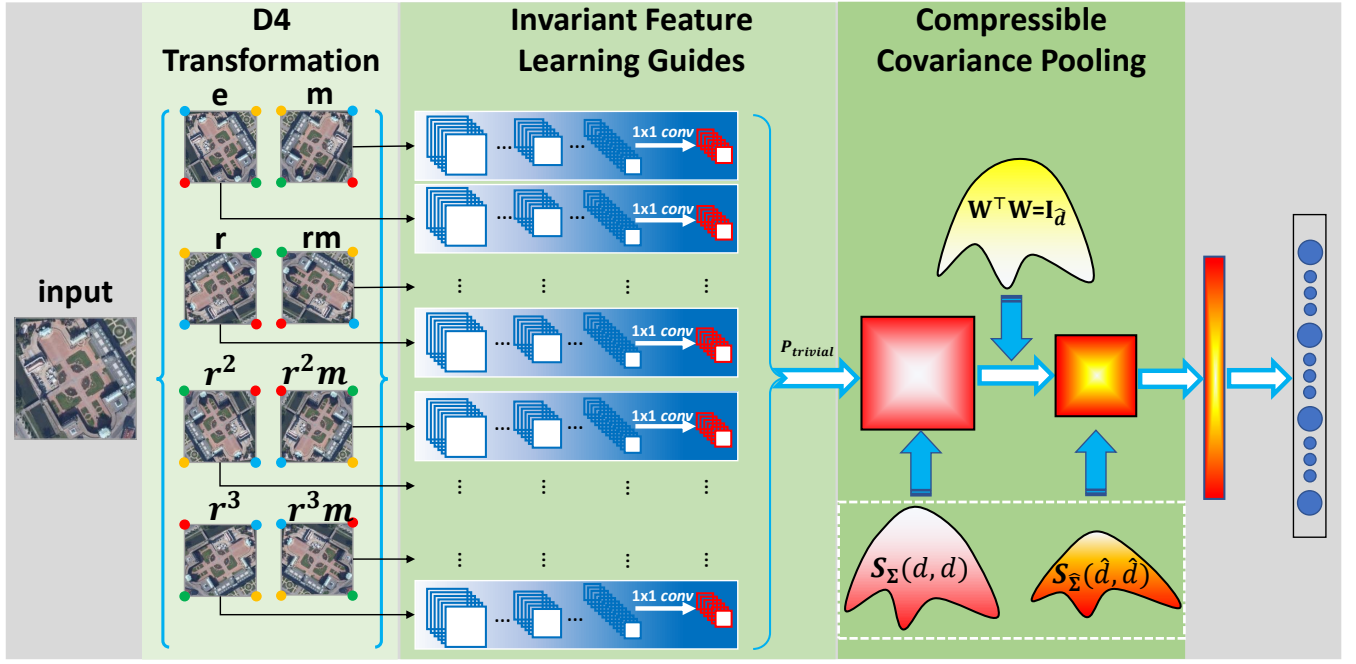
Fig. 2. An overview of the proposed IDCCP architecture. Given an input image, it will be used to generate multiple copies according to the D4 principle. Then, each copy will be fed into a subnetwork of Siamese-style CNNs to extract feature (Note:*1×1 conv* is only adopted in the Siamese architecture with ResNet50 as the backbone). $P_{trivial}$ is the projection layer to produce a trivial representation. Subsequently, orthogonal weights are adopted to compress high-dimensional manifold $S_\Sigma(d, d)$ to a compact manifold $S_{\hat\Sigma}(\hat d, \hat d)$. The resulting features will be flattened and fed into the classifier to generate predictions.

extraction. The naive way is that we adopt as many CNN networks as the order of the D_4 group. However, this method will exponentially increase the computational burdens. To address this problem, we exploit a Siamese-style architecture for feature extraction, which allows the weights to be shared among all subnetworks. To show how it works for preserving group structure, we provide the following proposition and the corresponding proof.

Proposition 1. *Let $\mathcal{X}$ be a set of images with the structure of symmetry square dihedral D_4 group, so $D_4 = \langle r, m : r^4 = m^2 = e, rm = mr^{-1} \rangle$ and let $\Phi : Siam(\mathcal{X}) \to \mathcal{F}$ be the feature extraction function. Then, the resulting features $\mathcal{F}$ will be given in the structure of the D_4 group.*

Proof. Let $T_g(\mathcal{X})$ be an action result of input D_4 group image and K be the convolution kernel of general CNN. The convolution operation on a 2-D image can be denoted as:

$$[T_g(\mathcal{X}) * K](i,j) = \sum_u \sum_v T_g(\mathcal{X})(u,v)K(i-u, j-v),$$
(2)

Then, we can exploit $u \to u+t, v \to v+t, u \to -u, v \to -v$, and $(u,v) \to r(u,v)$ (i.e., the substitution does not change the summation bounds since rotation is a symmetry of the sampling grid) to prove the relationships between convolution and translation, flip, and rotation, respectively. Results are:

$$[\Phi_t T_g(\mathcal{X})] * K(i,j) = \Phi_t[T_g(\mathcal{X}) * K](i,j)$$
$$[\Phi_m T_g(\mathcal{X})] * K(i,j) = \Phi_m[T_g(\mathcal{X}) * \Phi_{-m}K](i,j),$$
$$[\Phi_r T_g(\mathcal{X})] * K(i,j) = \Phi_r[T_g(\mathcal{X}) * \Phi_{r^{-1}}K](i,j)$$
(3)

□

A similar visual proof of the abovementioned relationships between convolution and transformations can be found in [8]. Furthermore, the pooling function that exists in CNN architecture has been proven to be commuted with the group action [6]. Hence, if an ordinary Siamese-style CNN learns transformed copies of the input image, the stack of feature maps will attain the same group structure as the transformed copies. It must be emphasized that the orientations of rotation may appear in either clockwise or counterclockwise depending on the implementation environment. If we let T_g and T'_g be actions on the sets of $\mathcal{X}$ and $\mathcal{F}$ that satisfy $T_{g_1 g_2} = T_{g_1} \bullet T_{g_2}$ and $T'_{g_1 g_2} = T'_{g_1} \bullet T'_{g_2}$, the transformations T_g and T'_g will induce actions $\mathbf{T}_g$ and $\mathbf{T}'_g$ on the space of $\mathcal{X}$ and $\mathcal{F}$. The difference between two spaces of $\mathcal{X}$ and $\mathcal{F}$ is the space field rather than the group structure. Thus, the transformation group of the input image can be preserved by using the Siamese-style CNNs.

B. Invariant Feature Learning Guides

Learning invariant features, as a particular case of learning equivariant features, is essential for many recognition tasks. It turns out that adopting a Siamese-style architecture can preserve the structure of the predefined transformations of inputs $\mathcal{X}$. The next step is to find the invariant subspace from the generated feature space $\mathcal{F}$. Because we assume that $\rho(g)$ are all orthogonal representations, it means that they are also unitary representations that cannot be decomposed, thus enabling us to derive invariant subspaces from the perspective of irreducible representations. Taking the D4 group as an example, its irreducible representations have been summarized

TABLE I
THE IRREDUCIBLE REPRESENTATIONS OF THE ROTO-REFLECTION D4 GROUP [7].

Irrep.	e	r	r^2	r^3	m	mr	mr^2	mr^3
$\rho_{1,1}$	[1]	[1]	[1]	[1]	[1]	[1]	[1]	[1]
$\rho_{1,-1}$	[1]	[1]	[1]	[1]	[-1]	[-1]	[-1]	[-1]
$\rho_{-1,1}$	[1]	[-1]	[1]	[-1]	[1]	[-1]	[1]	[-1]
$\rho_{-1,-1}$	[1]	[-1]	[1]	[-1]	[-1]	[1]	[-1]	[1]
ρ_2	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$

in TABLE. I where the orthogonality of the characters of representations can be verified.

Considering the fact that orthogonal representation is a real analog of unitary representation, the whole representation space can be formed by calculating the direct sum of all irreducible representations. For example, given a representation ρ, it can be decomposed by $\rho \simeq \lambda_1\tau_1 \oplus \lambda_2\tau_2 \oplus \ldots \lambda_T\tau_T$. As the characteristic function of ρ has been defined as $\mathcal{X}_\rho(g) = \text{tr}(\rho(g))$ with the matrix form $\rho(g)$ of ρ, the corresponding coefficients can be computed by using $\lambda_t = \frac{1}{|G|}\sum_{g \in G} \mathcal{X}_\rho(g)\mathcal{X}_{\tau_t(g)}$. The operator that projects ρ to $n_t\tau_t$ can be achieved by following $P_{\tau_t} = \dim(\tau_t)\sum_{g \in G} \overline{\mathcal{X}_{\tau_t}(g)}\rho(g)$. Since $\mathcal{X}_1(g) = 1$, we can obtain the trivial representation by calculating the average of $\rho(g)$:

$$P_{trivial} = \frac{1}{|G|}\sum_{g \in G}\rho(g). \qquad (4)$$

When we adopt the above trivial representation to train the classifier, the learned weights lie in the subspace of the entire action space (i.e., the average of all $\rho(g)$ is a subspace that is invariant to T-actions). To reveal the role of learning the trivial representation, we give the following theorem.

Theorem 1. *Given an input sample space $\mathcal{S} = \mathcal{X} \times \mathcal{Y} = \{(x_n, y_n)\}_{n=1}^{N} \in \mathbb{R}^d$, which is structured by a set of orthogonal transformation group G. Then the solution of minimizing the L2 regularized convex loss function:*

$$\min_{w,b} \frac{1}{N}\sum_{n=1}^{N} l\left(\langle w^\top x_n + b\rangle_{\mathbb{R}}, y_n\right) + \frac{\lambda}{2}||w||^2 \qquad (5)$$

lies in a vector subspace that is G-invariant, and the general error of the algorithm may be up to a factor $\sqrt{T}$ smaller than the general error of a non-invariant learning algorithm.

Proof. The proof of G-invariant has been given by [50] from the irreducible representation in the complex space, while Sokolic *et al.* [51] exploited a covering number to prove the general error of the invariant algorithm. □

For more details, we refer readers to [50] and [51] and reference herein. This theorem also induces essential properties of the trivial representation. Formally, for all $g \in G$, we can have:

$$\rho(g)w = w \Leftrightarrow \rho(g)w \subseteq w \text{ and } P_{trivial}w = w \qquad (6)$$

TABLE II
TENSOR PRODUCT OF IRREDUCIBLE REPRESENTATION OF THE
ROTO-REFLECTION D4 GROUP [50].

Irrep.	$\rho_{1,1}$	$\rho_{1,-1}$	$\rho_{-1,1}$	$\rho_{-1,-1}$	ρ_2
$\rho_{1,1}$	$\rho_{1,1}$	$\rho_{1,-1}$	$\rho_{-1,1}$	$\rho_{-1,-1}$	ρ_2
$\rho_{1,-1}$	$\rho_{1,-1}$	$\rho_{1,1}$	$\rho_{-1,-1}$	$\rho_{-1,1}$	ρ_2
$\rho_{-1,1}$	$\rho_{-1,1}$	$\rho_{-1,-1}$	$\rho_{1,1}$	$\rho_{1,-1}$	ρ_2
$\rho_{-1,-1}$	$\rho_{-1,-1}$	$\rho_{-1,1}$	$\rho_{-1,1}$	$\rho_{1,1}$	ρ_2
ρ_2	ρ_2	ρ_2	ρ_2	ρ_2	$\rho_{1,1}\oplus\rho_{1,-1}\oplus \rho_{-1,1}\oplus\rho_{-1,-1}$

The aforementioned theorem proves the G-invariance of augmented space contributes to reducing the general error of the learning algorithm but neglects to handle the massive parameters of the learning algorithm and the high-dimensional feature space. Instead, we deploy the learning algorithm to a shared-weights Siamese-style network and supply an effective compressible tensor representation in the following section.

C. Compressible Covariance Pooling

Covariance pooling, as a form of the second-order statistics feature, aims to establish the correlation between the spatial and channels of local CNN features to aggregate more distinguishing information. Suggested by [61], [55], and [60], we perform the second-order pooling in the form of a scatter covariance matrix:

$$\Sigma = \frac{1}{\text{hw}}\sum_{i=1}^{\text{hw}} P\left((\mathbf{f}_i - \bar{\mathbf{f}})(\mathbf{f}_i - \bar{\mathbf{f}})^\top\right) = \frac{1}{\text{hw}}P_{trivial}\left(\mathbf{F}\bar{\mathbf{I}}\mathbf{F}^\top\right).$$
$$(7)$$

where w and h are feature width and height. $P_{trivial}$ is projection function that we have introduced before. $\bar{\mathbf{f}} = \frac{1}{\text{hw}}\sum_{i=1}^{\text{hw}}\mathbf{f}_i$ is the mean of feature vectors. $\bar{\mathbf{I}} = \mathbf{I} - \frac{1}{\text{hw}}\mathbf{1}\mathbf{1}^\top \in \mathbb{R}^{\text{hw}\times\text{hw}}$ is the centering matrix, where $\mathbf{I}$ and $\mathbf{1}$ denote the identity matrix and the all-ones matrix, respectively.

Since the projection function $P_{trivial}$ is employed in the tensor space, the tensor product representation needs to be given concerning the irreducible representation in the D4 group. According to the distributive property of tensor product representation (e.g., given two representations ρ and σ, it satisfies $(\rho_1 \oplus \rho_2) \otimes (\rho_3 \oplus \rho_4) = (\rho_1 \otimes \rho_3) \oplus (\rho_2 \otimes \rho_3) \oplus (\rho_1 \otimes \rho_4) \oplus (\rho_2 \otimes \rho_4)$ and $\mathcal{X}_{\rho\otimes\sigma}(g) = \mathcal{X}_\rho(g)\mathcal{X}_\sigma(g)$, we can calculate the tensor product representations of irreducible representations. Combining the fact that the tensor product of irreducible representation and 1-D representation is irreducible, it allows us to decompose tensor products of D4 group

and present the results in TABLE II. For verifying the results, we take two representations $\rho(e)$ and $\rho(m)$ in TABLE I as an example, and the corresponding tensor product representations become 4-D vectors such that $\rho(e) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ and

$\rho(m) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$, respectively.

The obtained covariance matrix can be regarded as a form of representation, which is capable of capturing more information than the ordinary first-order statistical feature. However, its shortcomings are also obvious. The first and foremost drawback of such covariance pooling is its high dimensionality. Taking VGG architecture [62] as an example, the dimension of the vectorized covariance matrix generated from the last convolution layer will be 2^{18}. Rank deficiency is another weakness of covariance pooling because the number of CNN channels is much larger than the product of feature height and width.

The abovementioned reasons promote us to discover a compact form of covariance pooling. Considering that the covariance matrix is an SPD matrix, it is necessary to retain the geometry of the SPD manifold while reducing the matrix dimension. To accomplish this goal, we provide a method based on the following proposition.

Proposition 2. *Let* $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$ *be the covariance matrix generated from the last convolution layer and* $\mathbf{W} \in \mathbb{R}^{d \times \hat{d}}$ *be an orthogonal, row full rank matrix with* $\hat{d} < d$. *Then, the bilinear form of transformation matrix* $\mathbf{W}$ *maps* $\mathbf{\Sigma}$ *to a valid SPD matrix* $\hat{\mathbf{\Sigma}} \in \mathbb{R}^{d \times \hat{d}}$.

Proof. The bilinear mapping function can be generally denoted as $\mathcal{B} : \mathbf{\Sigma} \times \mathbf{W} \to \hat{\mathbf{\Sigma}}$. In order to express it more accurately, we can rewrite it in the form of: $\hat{\mathbf{\Sigma}} = \mathbf{W}^\top \mathbf{\Sigma} \mathbf{W}$. Due to the orthogonality and row full rank of transformation matrix $\mathbf{W}$, the elements generated by transformation weights are naturally located on a non-compact Stiefel manifold $\mathcal{S}^*(\hat{d}, d) \triangleq \left\{ \mathbf{W} \in \mathbb{R}^{d \times \hat{d}} : \mathbf{W}^\top \mathbf{W} = \mathbf{I}_{\hat{d}} \right\}$ and can be transformed into a compact manifold $\mathcal{S}(\hat{d}, d)$. Then, the resulting matrix $\hat{\mathbf{\Sigma}} \in \mathbb{R}^{\hat{d} \times \hat{d}}$ is a valid but very compact SPD matrix because $\hat{d} < d$. $\square$

The abovementioned claim and proof are trivial, but it guides us to convert those high-dimensional SPD matrices $\mathbf{\Sigma}$ to new, low-dimensional SPD matrices $\hat{\mathbf{\Sigma}}$ with $\hat{d} < d$, $\hat{\mathbf{\Sigma}} \in Sym_{\hat{d}}^+$. Compared with most existing methods that directly map SPD manifold into the Euclidean space [52], [54], [57], [56], [61], [60], our method can certainly preserve the inherent manifold structure of high-dimensional SPD matrices. However, given a non-compact Stiefel manifold, a matrix form of writing linearly independent column vectors (i.e., $\hat{d}$-frames), has no closed-form of geodesic curves. In other words, it is infeasible to optimize on the manifold directly [63]. The relatively tractable strategy is to endow non-compact Stiefel manifold with a pseudo-Riemannian manifold so that

the gradient of geodesic distance can be derived from a smooth manifold and present in a closed form. To achieve this target, we impose the orthogonal constraints on $\mathbf{W}$ (precisely speaking, it is semi-orthogonal matrix under this scenario). Consequently, the entities of transformation weight $\mathbf{W}$ reside on a compact Stiefel manifold $\mathcal{S}(\hat{d}, d)$, which allows us to find the optimal solutions of the weight matrix.

Furthermore, the abovementioned function for feature dimension reduction can also be regarded as an intertwining operator when we impose orthogonal constraints $\mathbf{W}^\top \mathbf{W} = \mathbf{I}_{\hat{d}}$ on transformation weight. Recalling the introduction of intertwining in preliminaries, the produced projection space is also the representation space. Thus, the low-dimensional representation can be achieved by imposing low-rank constraints on weight $\mathbf{W}$. Specifically, we can first line up the eigenvalues of $\mathbf{\Sigma}$ by employing eigenvalue decomposition function and then find the elements with the larger variance to retain. However, matrix decomposition often requires more computational costs and time-consuming [61], [3]. Rather than using cumbersome decomposition functions, the bilinear mapping function can transform the input SPD matrix into a new, low-dimensional SPD matrix that is useful for subsequent optimization.

D. Invariant Classifier Training

The compressible covariance pooling method has been described in the last section, which maps the high-dimensional manifold to a low-dimensional compact manifold. Different from the mainstream methods, our algorithm deduces a rank efficient representation on manifold space while retaining the inherent manifold structure.

The elements of the resulting low-dimensional SPD matrices reside on the Riemannian manifold, which needs to be transformed into the Euclidean space so that the distance between different elements can be measured by the Euclidean operations. The natural choice is to employ the logarithm of SPD matrices since it reflects the true geodesic distance of the manifold. Furthermore, the logarithm of an SPD matrix will give rise to the matrix with a Lie group, and then, all Euclidean operations can be adopted. However, the logarithm will change the magnitude order of small eigenvalues and usually not robust in practice [61], [54]. Instead, we are committed to learning more robust square root normalization of matrices, which can be considered as the approximate Riemannian geometry in covariance matrices [61].

It is well-known that any SPD matrix has a unique square root, which can be obtained by using SVD or EIG. Although SVD or EIG yield the accurate solution of the square root of a matrix, they are time-consuming and often cannot be well-supported by GPU acceleration [61], [3]. Inspired by [60], we adopt the iSQRT-COV approach that uses a variation of the Newton method to iteratively calculate the square root of the matrix. Especially, given $\mathbf{C}_0 = \frac{\hat{\mathbf{\Sigma}}}{\text{tr}(\hat{\mathbf{\Sigma}})}$ and $\mathbf{D}_0 = \mathbf{I}$, the Newton-Schulz method [60] allows us to compute the square root $\mathbf{C}$ of $\hat{\mathbf{\Sigma}}$ by using the following iterations:

$$\begin{aligned} \mathbf{C}_j &= \frac{1}{2}\mathbf{C}_{j-1}(3\mathbf{I} - \mathbf{D}_j \mathbf{C}_{j-1}), \\ \mathbf{D}_j &= \frac{1}{2}(3\mathbf{I} - \mathbf{D}_j \mathbf{C}_{j-1})\mathbf{D}_j, \end{aligned} \quad (8)$$

where $j = 1, \ldots, J$ is the iteration steps. With the condition of $||\hat{\boldsymbol{\Sigma}} - \mathbf{I}|| < \frac{1}{2}$, $\mathbf{C}_j$ and $\mathbf{D}_j$ are guaranteed to quadratically converge to $\mathbf{C}^{\frac{1}{2}}$ and $\mathbf{C}^{-\frac{1}{2}}$, respectively. Briefly, it means that $\mathbf{C}^2 = \hat{\boldsymbol{\Sigma}}$ and $\mathbf{C} = \boldsymbol{\Psi} \boldsymbol{\Lambda} \boldsymbol{\Psi}^\top$ described in EIG format, where $\mathbf{C} = \boldsymbol{\Psi}$ is an orthogonal matrix and $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1^{\frac{1}{2}}, \boldsymbol{\lambda}_2^{\frac{1}{2}}, \ldots, \boldsymbol{\lambda}_{d'}^{\frac{1}{2}})$ is a diagonal matrix.

Once the square-root of the SPD matrix is obtained, we can adopt the Euclidean operations to measure the distance of elements on the flatted Stiefel Manifold. Considering the fact that the initialization of $\mathbf{C}_0$ has changed the magnitude of the matrix value, we then use $\hat{\mathbf{C}} = \sqrt{\operatorname{tr}(\hat{\boldsymbol{\Sigma}})} \mathbf{C}_j$ to counteract such changes [60]. The resulting matrix $\hat{\mathbf{C}}$ can be used to train the classifier. Let us suppose that $\hat{\mathbf{W}}$ be the corresponding weight matrix of $\hat{\mathbf{C}}$. The objective function in **Theorem 1.** can be rewritten by substituting w with $\hat{\mathbf{W}}$, and then yield the following expression:

$$
\begin{aligned}
&\min_{\hat{\mathbf{W}}, b} \frac{1}{N} \sum_{i=1}^{N} l \left(\left\langle \operatorname{tr} \left(\hat{\mathbf{W}}^\top \hat{\mathbf{C}} \right) + b \right\rangle_{\mathbb{R}}, y_i \right) + \frac{\lambda}{2} ||\hat{\mathbf{W}}||^2 \\
&= \min_{\hat{\mathbf{W}}, b} \frac{1}{N} \sum_{i=1}^{N} l \left(\left\langle \operatorname{tr} \left(\hat{\mathbf{W}}^{(1)} \hat{\mathbf{C}}^{(1)} \right) + b \right\rangle_{\mathbb{R}}, y_i \right) + \frac{\lambda}{2} ||\hat{\mathbf{W}}||^2.
\end{aligned}
\tag{9}
$$

For brevity, we omit the transpose operator $\top$ in the last line since $\hat{\mathbf{W}} = \hat{\mathbf{W}}^\top$. The final result highlights the key advantage of our classifier, which avoids the direct optimization on the original high-dimensional weights $\mathbf{W}$.

E. Back-propagation

Stochastic gradient descent (SGD), as one of the most popular gradient calculation algorithms, is widely adopted for training deep CNNs. In our scenario, we employ SGD to compute the gradient of the given objective function with respect to the transformation matrix $\mathbf{W}$ and the second-order statistical feature $\boldsymbol{\Sigma}$. Let us write the derivative of $\hat{\mathbf{C}}$ as $\left(\frac{\partial l}{\partial \hat{\mathbf{C}}} \right)$ that derives from the Softmax layer. Then, we can use the following chain rules to calculate the matrix derivatives:

$$
\begin{aligned}
\operatorname{tr} \left(\left(\frac{\partial l}{\partial \hat{\mathbf{C}}} \right)^\top d\hat{\mathbf{C}} \right) &= \operatorname{tr} \left(\left(\frac{\partial l}{\partial \hat{\mathbf{C}}_J} \right)^\top d\hat{\mathbf{C}}_J + \left(\frac{\partial l}{\partial \hat{\boldsymbol{\Sigma}}} \right)^\top d\hat{\boldsymbol{\Sigma}} \right), \\
\operatorname{tr} \left(\left(\frac{\partial l}{\partial \mathbf{W}} \right)^\top d\boldsymbol{\Sigma} \right) &= \operatorname{tr} \left(\left(\frac{\partial l}{\partial \hat{\boldsymbol{\Sigma}}} \right)^\top d\hat{\boldsymbol{\Sigma}} + \left(\frac{\partial l}{\partial \mathbf{W}} \right)^\top d\boldsymbol{\Sigma} \right),
\end{aligned}
\tag{10}
$$

where $d\hat{\mathbf{C}}$ is the variation of $\hat{\mathbf{C}}$. According to expression at the first line, we can derive the derivative of $\hat{\boldsymbol{\Sigma}}$ through some manipulations. For more details, we refer readers to [60] and reference it herein. Once we obtained $\frac{\partial l}{\partial \hat{\boldsymbol{\Sigma}}}$, we can exploit it to compute the gradient for updating $\mathbf{W}$.

As described in IV-C, we project all elements on the Stiefel manifold $\mathcal{S}(\hat{d}, d)$ into the Euclidean space so that we can use the Euclidean operations to measure the distance between projected elements. However, directly using the back propagation rules in the Euclidean space to calculate the gradient of the Stiefel manifold cannot guarantee that the orthogonality of weights $\mathbf{W}$. To this end, we introduce the Euclidean inner product in the tangent space of the Stiefel manifold as a

new strategy for updating the gradient of our covariance pooling. Therefore, the Stiefel manifold is transformed into a Riemannian manifold so that we can borrow the method of optimizing the Riemannian manifold to calculate the gradient of the Stiefel manifold. When employing the Euclidean inner product, the corresponding gradient of the current points $\mathbf{W}^t$ on the Riemannian manifold $G_e^{\mathcal{S}} l(\mathbf{W}^t)$ can be obtained by:

$$
G_e^{\mathcal{S}} l(\mathbf{W}^t) = \frac{\partial l}{\partial \mathbf{W}^t} - \mathbf{W}^t \left(\frac{\partial l}{\partial \mathbf{W}^t} \right)^\top \mathbf{W}^t,
\tag{11}
$$

where $\frac{\partial l}{\partial \mathbf{W}^t}$ is the normal component of the gradient in the Euclidean space, which can be obtained by using the second expression of Eq.(10) as:

$$
\frac{\partial l}{\partial \mathbf{W}^t} = 2 \frac{\partial l}{\partial \hat{\boldsymbol{\Sigma}}} \mathbf{W}^t \boldsymbol{\Sigma},
\tag{12}
$$

When we obtain the Riemannian gradient, we need to seek the descent direction of the gradient (i.e., we use steepest gradient descent in our scenario) and ensure that the new update points $\mathbf{W}^{t+1}$ are located on the Stiefel manifold. To achieve this, we adopt the QR-decomposition retraction $Z_{\mathbf{W}}(\xi) = qf(\mathbf{W} + \xi)$ introduced in [64], [65], [66]. Here, $qf(\cdot)$ is the adjusted Q factors of the QR-decomposition and R factors in an upper triangular matrix with strictly positive elements on the diagonal. Thus, the decomposition is guaranteed to be unique and orthogonal. Through defining the learning rate as η, we can compute the new point by:

$$
\mathbf{W}_{t+1} = qf \left(\mathbf{W}_t - \eta G_e^{\mathcal{S}} l(\mathbf{W}^t) \right),
\tag{13}
$$

Once the derivation of $\frac{\partial l}{\partial \boldsymbol{\Sigma}}$ has been achieved, we can derive the derivative for the input feature $\mathbf{F}$ with using:

$$
\frac{\partial l}{\partial \mathbf{F}} = \left(\frac{\partial l}{\partial \boldsymbol{\Sigma}} \left(\frac{\partial l}{\partial \boldsymbol{\Sigma}} \right)^\top \right) \hat{\mathbf{I}} \mathbf{F}.
\tag{14}
$$

V. EXPERIMENTS

A. Datasets Description

To evaluate the effectiveness of the proposed method, we carried out comprehensive experiments on four publicly available aerial image datasets. Experimental data sets include UC Merced Land-Use dataset [18], Aerial Image dataset [1], NWPU-RESISC45 dataset [2], and a recently released OPTIMAL-31 datasets [67]. The statistics of datasets have been summarized in TABLE III. In addition to the information listed in TABLE III, the spatial resolution of aerial images is another important factor that makes aerial images differ from ordinary scene images. Especially, the spatial resolution of the UC Merced Land-Use dataset [18] is about 0.3m/pixel, while it becomes 0.5-8m/pixel on AID [1], and it is even more diversified on NWPU-RESISC45 dataset [2], which is about 0.2-30m/pixel. Through comparing with the information described in Section I, we can see that most of the data sets used to evaluate our model can satisfy the definition of high spatial resolution images.

TABLE III
THE STATISTICS OF REMOTE SENSING SCENE DATASETS.

Datasets	No. Images	No. Class	No. Images (Per-class)	Image Size
UC Merced Land-Use [18]	2,100	21	100	256×256
AID [1]	10,000	30	220∼420	600×600
NWPU-RESISC45 [2]	31,500	45	700	256×256
OPTIMAL-31 [67]	1,861	31	60	256×256

B. Implement Details

We implement our method using the GPU version of Tensorflow in v1.10.0. We construct two Siamese-style architectures based on two standard CNNs, including VGGNet [62] and ResNet50 [68]. We remove all fully-connected layers from the original backbone networks and then insert our projection layer and compressible covariance pooling layer at the same place to train the invariant classifier. The batch size is set to 32 during training. We employ SGD with a momentum of 0.9 and a weight decay of 0.0005 to optimize the gradient. The initial learning rate is set to 0.1 and becomes 0.01 when fine-tuning the entire network. We employ exponential decay in the training process, with a decay factor of 0.9 in every 10 epochs. We adopt five-fold cross-validation to reduce the influence of the randomness and obtain reliable results. When we train our model on the UC Merced Land-Use dataset [18], the NWPU-RESISC45 dataset [2], and the OPTIMAL-31 dataset [67], we randomly crop patches of 224×224 pixels from the input image and flip them horizontally or vertically. During the test, the manipulation of central cropping is adopted to obtain patches of the same size as in training. These operations are also applied to AID [1], but the size of patches becomes 448×448 pixels.

C. Analysis of Model Complexity

In view of the success of bilinear pooling [52] and its relevance to our method, we compared the differences of two models in various aspects and listed the results in TABLE IV. Especially, the aspects of comparison include input feature dimension, complexity and corresponding parameter size, classifier complexity and its parameter size, and overall model parameters. In order to show the function of the projection layer, all results are obtained by employing the Siamese-style architecture based on ResNet50 [68]. With using the projection layer, the feature dimension can be reduced to the same par with the last convolution layer in VGGNet [62]. As shown in TABLE IV, our invariant deep compressible covariance pooling (IDCCP) model requires an additional 4-MB feature parameter compared to the bilinear pooling model [52]. However, this operation is more conducive to reducing the feature dimension and, thus, greatly reducing the number of classifier parameters. Namely, our IDCCP model not only learns compressible feature representations but also enables us to train more compact classifiers.

D. Comparison with State-of-the-Arts

We provide four variants of the IDCCP model and display the classification results in TABLE V. It is plain to see that

our IDCCP models achieved extremely competitive results on all experimental datasets. In particular, the performance of the IDCCP model based on ResNet50 [68] is superior to the latest MG-CAP model [3] on all datasets and even far exceeds baseline methods (e.g., our method is improved by about 10% compared with the standard method of AlexNet + SVM on the challenging NWPU-RESISC45 dataset [2]). When using VGGNet [62], the MG-CAP model [3] shows strong competitiveness in classification accuracy, but even if GPU acceleration is enabled, it requires 4.5 times the number of transformations and nearly seven times in terms of inference time. When we employ ResNet50-based Siamese-style architecture, the proposed IDCCP models can obtain accuracy rates higher than 91% and 93% under two split ratios on the NWPU-RESISC45 dataset [2], respectively. On AID [1], we can obtain 94.80±0.18 with using 20% training samples, which exceeds the best results of MG-CAP model [3] and DCNN model [69] by 1.46% and 3.98%, respectively. Under the 50% training ratio, the GCFs+LOFs model [72] presents surprisingly better than most existing methods but still below the optimal level of our IDCCP framework. Similarly, on UC Merced Land-Use dataset [18], our IDCCP achieves the highest accuracy among the listed algorithms. In addition, we show the comparisons of our IDCCP model with previous methods on the OPTIMAL-31 dataset [67]. As shown in TABLE VI, three variants of our proposed method achieve higher results than the latest algorithm. Even our worst IDCCP model can still exceed the result of fine-tuning AlexNet by more than 10%. By using ResNet50 architecture, our IDCCP model can improve the optimal performance of ARCNet-VGGNet16 by 1.84%. These indicate that the classification performance can be improved by incorporating the prior knowledge of the input image.

Generalization ability is vitally important for measuring the effectiveness of deep learning models. By analyzing the data listed in Table V, it is not difficult to see that the variants of our IDCCP model can always bring relatively stable benefits to different data sets. Concretely, using different proportions of training data on the NWPU-RESISC45 [2] (i.e., 10% versus 20% training ratios), the difference between our IDCCP model is about 2%, but this gap is significantly enlarged on other models (e.g., about 4% by CapsNet with VGGNet [73] and about 3% by MSCP with AlexNet or VGGNet [70]). A similar degree of gain is also reflected in the AID [1] with different partitions. However, most of the existing methods are not stable enough under different partitions, including DCNN [69] (about 6%-9%), GCFs+LOFs [72] (about 4%), and SVM-based methods [69] (about 7%-9%). It is worth noting that the actual number of samples corresponding to different training ratios on two different datasets (10% and 20% on NWPU-RESISC45 [2] versus 20% and 50% on AID [1]) is in the same order of magnitude (3,150 on NWPU-RESISC45 [2] versus 3,000 on AID [1]). Therefore, similar gains in different scenarios also reflect that the robustness of our IDCCP model.

Through comparing the variants of our IDCCP model, we found that the IDCCP model based on VGGNet [62] has achieved very competitive results on all experimental datasets. Especially, using VGGNet [62], our IDCCP model can obtain comparable results to the similar methods, such

TABLE IV

COMPARISON WITH THE BILINEAR POOLING METHOD [52] IN TERMS OF FEATURE DIMENSIONALITY, COMPUTATIONAL COMPLEXITY AND THE NUMBER OF PARAMETERS (RESNET50 [68]-BASED SIAMESE-STYLE ARCHITECTURE). WHERE $d_p = 512$ AND K DENOTE THE PROJECTION LAYER AND THE NUMBER OF CATEGORIES, RESPECTIVELY. (W/ AND W/O INDICATE WITH PROJECTION LAYER AND WITHOUT PROJECTION LAYER, RESPECTIVELY.)

Methods	Feature Dim.	Feature Comp.	Classifier Comp.	Feature Param.	Classifier Param.	Model Param.
Bilinear pooling [52] (w/o d_p)	d^2 [4,194K]	$O(hwd^2)$	$O(Kd^2)$	0	Kd^2 [$K\cdot 16$MB]	[118MB]
Our IDCCP-512 (w/ d_p)	$\hat{d}_1^2$ [256K]	$O(hwd_pd + hw\hat{d}_1^2)$	$O(K\hat{d}_1^2)$	dd_p [4MB]	$K\hat{d}_1^2$ [$K\cdot 1$MB]	[30MB]
Our IDCCP-64 (w/ d_p)	$\hat{d}_2^2$ [4K]	$O(hwd_pd + hw\hat{d}_2^2)$	$O(K\hat{d}_2^2)$	dd_p [4MB]	$K\hat{d}_2^2$ [$K\cdot 16$KB]	[25MB]

TABLE V

COMPARISON WITH STATE-OF-THE-ART APPROACHES IN TERMS OF OVERALL ACCURACY AND STANDARD DEVIATION (%). T.R. IS THE ABBREVIATION OF TRAINING RATIO.

Methods	NWPU-RESISC45 [2]		AID [1]		UC-Merced Land-Use [18]	Publication Year
	T.R.=10%	T.R.=20%	T.R.=20%	T.R.=50%	T.R.=80%	
AlexNet+SVM [69]	81.22±0.19	85.16±0.18	84.23±0.10	93.51±0.10	94.42±0.10	2018
GoogLeNet+SVM [69]	82.57±0.12	86.02±0.18	87.51±0.11	95.27±0.10	96.82±0.20	2018
VGGNet+SVM [69]	87.15±0.45	90.36±0.18	89.33±0.23	96.04±0.13	97.14±0.10	2018
MSCP with AlexNet [70]	81.70±0.23	85.58±0.16	88.99±0.38	92.36±0.21	97.29±0.63	2018
MSCP+MRA with AlexNet [70]	88.31±0.23	87.05±0.23	90.65±0.19	94.11±0.15	97.32±0.52	2018
MSCP with VGGNet [70]	85.33±0.17	88.93±0.14	91.52±0.21	94.42±0.17	98.36±0.58	2018
MSCP+MRA with VGGNet [70]	88.07±0.18	90.81±0.13	92.21±0.17	96.56±0.18	98.40±0.34	2018
DCNN with AlexNet [69]	85.56±0.20	87.24±0.12	85.62±0.10	94.47±0.10	96.67+0.10	2018
DCNN with GoogLeNet [69]	86.89±0.10	90.49±0.15	88.79±0.10	96.22±0.10	97.07+0.12	2018
DCNN with VGGNet [69]	89.22±0.50	91.89±0.22	90.82±0.16	96.89±0.10	98.93±0.10	2018
RTN with VGGNet [47]	89.90	92.71	92.44	-	98.96	2018
Two-Stream Fusion [71]	80.22±0.22	83.16±0.18	92.32±0.41	94.58±0.25	98.02±1.03	2018
GCFs+LOFs [72]	-	-	92.48±0.38	96.85±0.23	99.00±0.35	2018
CapsNet with VGGNet [73]	85.08±0.13	89.18 ± 0.14	91.63±0.19	94.74±0.17	98.81±0.22	2019
MG-CAP with Bilinear VGGNet [3]	89.42±0.19	91.72 ± 0.16	92.11±0.15	95.14±0.12	98.60±0.26	2020
MG-CAP with Log-E VGGNet [3]	88.35±0.23	90.94 ± 0.20	90.17±0.19	94.85±0.16	98.45±0.12	2020
MG-CAP with Sqrt-E VGGNet [3]	90.83±0.12	92.95 ± 0.13	93.34±0.18	96.12±0.12	99.00±0.10	2020
IDCCP with VGGNet-512(ours)	90.88±0.18	92.80±0.10	93.58±0.24	96.33±0.12	98.45±0.12	-
IDCCP with VGGNet-64(ours)	89.61±0.19	91.75±0.18	92.33±0.25	94.82±0.22	97.35±0.20	-
IDCCP with ResNet50-512(ours)	**91.55**±0.16	**93.76**±0.12	**94.80**±0.18	**96.95**±0.13	**99.05**±0.20	-
IDCCP with ResNet50-64(ours)	91.31±0.22	93.66±0.21	94.64±0.23	96.73±0.18	98.57±0.24	-

TABLE VI

COMPARISON WITH STATE-OF-THE-ART METHODS IN TERMS OF OVERALL ACCURACY AND STANDARD DEVIATION (%).

Method	OPTIMAL-31 [67]
	Training Ratio=80%
Fine-tuning AlexNet [67]	81.22 ± 0.19
Fine-tuning GoogLeNet [67]	82.57 ± 0.12
Fine-tuning VGGNet16 [67]	87.45 ± 0.45
ARCNet-Alexnet [67]	85.75 ± 0.35
ARCNet-ResNet34 [67]	91.28 ± 0.45
ARCNet-VGGNet16 [67]	92.70 ± 0.35
IDCCP with VGG-512 (ours)	93.82±0.32
IDCCP with VGG-64 (ours)	92.13±0.38
IDCCP with ResNet50-512 (ours)	**94.89**±0.22
IDCCP with ResNet50-64 (ours)	94.54±0.28

them are even only 0.1%. For example, using ResNet50 to train our model on the NWPU-RESISC45 dataset [2] (under 20% training ratio) can achieve 93.76% and 93.66% accuracy. Apart from the advanced structure of ResNet50, our IDCCP model also benefits from the orthogonal feature reduction layer and the projection layer (i.e., 1×1 convolution layer), which can effectively remove redundant feature information.

In addition to comparing overall accuracy, we also provide examples of confusion matrices to show category-level details. For saving space, we randomly selected the results of relatively difficult scenarios on each dataset and displayed in Fig. 3. It can be clearly seen that the darkest color blocks appear on the diagonals of all confusion matrices. On NWPU-RESISC45 V-D, there exists 40 categories among all 45 categories obtain a classification accuracy rate higher than 90%. Especially, it reports that the classification accuracy of the **Palace** and **Church** categories is 82% and 77%, respectively. These two categories are acknowledged as the most visually similar categories in the NWPU-RESISC45 dataset [2]. Our model can significantly improve the classification accuracy when comparing with the benchmark methods described in [2] (i.e., with improvements of 24% and 25%, 7% and 11% compared with VGGNet [62] and Fine-tuned VGGNet [62], respectively.). Besides, the accuracy that we obtained in these two categories is

as RTN [47] and MG-CAP [3], and even significantly better than MSCP [70]. The full-rank IDCCP model (i.e., VGGNet-512) obtained a classification accuracy rate of about 10% higher than the two-stream fusion model [71] on the NWPU-RESISC45 dataset [2]. Furthermore, at the expense of the accuracy of the tolerable range (i.e., about 1%-2%), we can compress the model parameters to 1/64 of the original second-order features. The performance gap between IDCCP models based on full-rank and low-rank is rarely small, and some of

(a) NWPU-RESISC45 [2] (10% training ratio)

(b) AID [1] (20% training ratio)

(c) UC Merced [18] (50% training ratio)
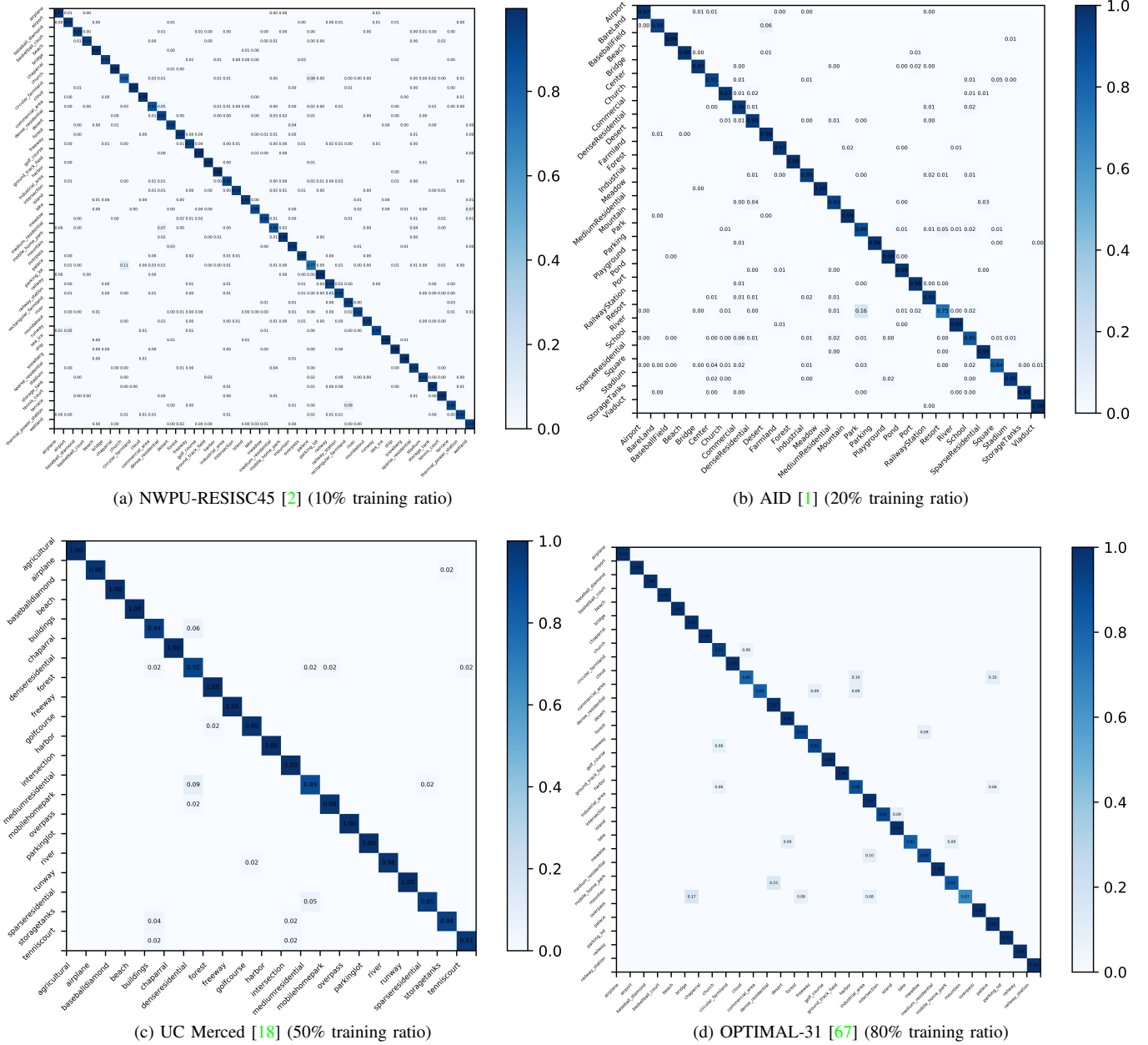
(d) OPTIMAL-31 [67] (80% training ratio)

Fig. 3. Results of the confusion matrix on different datasets achieved by our IDCCP model based on ResNet50-512 architecture (not cherry-picking).

higher than the two latest technologies by 3% and 4% [69], 2% and 1% [47], respectively. On AID V-D, the reported accuracy of **School** is 85%, which exceeds the algorithm introduced in [1] by 18%. Due to the high similarity, 16% of **Resort** images are misclassified as **Park**. The classification results of the easily confused **dense residences**, **medium-sized residences** and **sparse residences** are 96%, 93% and 99%, respectively. The sparsest confusion matrix V-D is obtained by evaluating the proposed framework on the UC-Merced Land-Use dataset [18] with the training ratio of 50%. Especially, the accuracy of the **medium residence** category is 89% (0.09% of images are misclassified as **intensive residence**, which is the only category with an accuracy of less than 90%. On the OPTIMAL-31 dataset [67], there are 20 categories of test data

that can be classified 100% correctly V-D, including those challenging categories, such as **Church** and **Island**.

E. Ablation Study and Analysis

1) Compactness and Effectiveness: In Table VII, extensive results are listed to show the effect of feature dimensionality and D4 transformation group on classification accuracy and a single image inference time. For a fair comparison, we ensure that all hyperparameters are consistent and then obtain the interaction time of a single image by calculating the ratio of the total test duration to the number of test samples. When we reduce the feature size, the gap in classification accuracy is not significantly enlarged. For example, with ResNet50 architecture [68], the accuracy only decreases by 0.86% even

TABLE VII
COMPARISON OF CLASSIFICATION ACCURACY AND SINGLE IMAGE
INFERENCE TIME. EXPERIMENTS WERE CONDUCTED ON
NWPU-RESISC45 DATASET [2] WITH USING 10% TRAINING SAMPLES.

Networks	Feature Dim.	Accuracy (%)		Time (sec/per image)	
		w/ D4	w/o D4	w/ D4	w/o D4
ResNet50 [68]	2048	-	90.02	-	0.0219
	512	91.64	90.05	0.0768	0.0105
	64	91.26	89.94	0.0744	0.0093
	16	90.78	89.83	0.0721	0.0087
VGGNet [62]	512	91.11	89.44	0.0324	0.0063
	64	89.78	88.34	0.0322	0.0059
	16	88.62	87.21	0.0317	0.0052
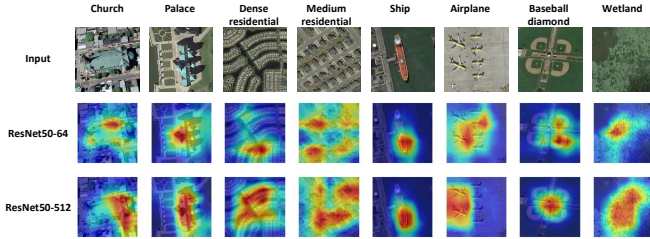
Fig. 4. Selected images for qualitative visualisation.

if we compress the feature space to 1/64 of the original feature space. It is worth noting that our IDCCP model allows features to be compressed into a very compact space (i.e., 16×16) without sacrificing too much accuracy. Interestingly, the classification accuracy is slightly improved when we adopt 1 × 1 convolution layer to map the CNN feature to a lower feature space. The reason for this phenomenon is that 1 × 1 convolution can reduce the diversity and redundancy of feature maps, thereby improving the discriminative power of learned feature [59]. Due to the limited capability of our PC, the accuracy of equipping the D4 transformation group has been omitted. However, this hardly affects us to investigate the effectiveness of the D4 transformation group. At the feature size of 16×16, the IDCCP model based on ResNet50 [68] achieved 89.9% accuracy, which can exceed the full-rank constrained VGGNet model. It not only influenced by the superior structure of ResNet50 [68] but also reflects the effectiveness of the projection layer. In addition, ResNet50 [68]-based IDCCP model, a single image inference time, only needs about 0.07 and 0.01 seconds for equipping or not equipping the D4 group, respectively. Due to the relatively shallow CNN structure, the inference time reduce by half when using VGGNet [62] architecture.

2) Qualitative Visualization: Through the comparison of the above experiments, we find that the overall accuracy of the compressed model can be kept at the same par with the uncompressed model. This prompted us to find the evidence from the interpretability of the model. As shown in. 4, we select example images from the NWPU-RESISC45 datasets [2] and show the corresponding heatmaps by utilizing the Grad-Cam algorithm [74]. When using ResNet50-64 architecture, we found that our model can focus on small patches that benefit to distinguish subtle differences between visually similar images, such as **Church** and **Palace**, **Dense residential** and

Medium residential. Compared with ResNet50-64, the area of attention map is significantly expanded when using ResNet50-512 model. Namely, it allows the model to capture more texture information and could be the reason why ResNet50-512 performs slightly better than ResNet50-64 model.

VI. CONCLUSION

In this article, we proposed a novel IDCCP model to handle the variations in the classification of aerial images. The model benefits from the use of Siamese CNNs to learn the trivial representation of the predefined transformation group. The obtained representation can be deployed to the scenarios of the second-order representation. Meanwhile, we endowed the weight matrix with the form of Stiefel manifold and employed it to reduce the dimensions of the SPD manifold. Finally, the resulting features are flattened to train invariant classifiers. In the future, we will focus on exploring the impact of more complex group structures.

VII. ACKNOWLEDGMENT

REFERENCES

[1] G.-S. Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, L. Zhang, and X. Lu, "Aid: A benchmark data set for performance evaluation of aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3965–3981, 2017. 1, 2, 7, 8, 9, 10

[2] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: Benchmark and state of the art," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1865–1883, 2017. 1, 2, 7, 8, 9, 10, 11

[3] S. Wang, Y. Guan, and L. Shao, "Multi-granularity canonical appearance pooling for remote sensing scene classification," *IEEE Transactions on Image Processing*, vol. 29, no. 1, pp. 5396–5407, 2020. 1, 2, 6, 8, 9

[4] J. Li, X. Huang, and J. Gong, "Deep neural network for remote-sensing image interpretation: Status and perspectives," *National Science Review*, vol. 6, no. 6, pp. 1082–1086, 2019. 1, 2

[5] P. Y. Simard, D. Steinkraus, J. C. Platt *et al.*, "Best practices for convolutional neural networks applied to visual document analysis." vol. 3, no. 2003, 2003. 2, 3

[6] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International conference on machine learning*, 2016, pp. 2990–2999. 2, 3, 4

[7] T. S. Cohen and M. Welling, "Steerable cnns," *arXiv preprint arXiv:1612.08498*, 2016. 2, 3, 5

[8] S. Dieleman, J. De Fauw, and K. Kavukcuoglu, "Exploiting cyclic symmetry in convolutional neural networks," *arXiv preprint arXiv:1602.02660*, 2016. 2, 3, 4

[9] J. F. Henriques and A. Vedaldi, "Warped convolutions: Efficient invariance to spatial transformations," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1461–1469. 2

[10] J. A. dos Santos and O. A. B. Penatti, "Evaluating the potential of texture and color descriptors for remote sensing image retrieval and classification." in *In VISAPP (2)*, 2010, pp. 203–208. 2

[11] H. Li, H. Gu, Y. Han, and J. Yang, "Object-oriented classification of high-resolution remote sensing imagery based on an improved colour structure code and a support vector machine," *International journal of remote sensing*, vol. 31, no. 6, pp. 1453–1470, 2010. 2

[12] O. A. Penatti, K. Nogueira, and J. A. dos Santos, "Do deep features generalize from everyday objects to remote sensing and aerial scenes domains?" in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 44–51. 2

[13] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002. 2

[14] J. Ren, X. Jiang, and J. Yuan, "Learning lbp structure by maximizing the conditional mutual information," *Pattern Recognition*, vol. 48, no. 10, pp. 3180–3190, 2015. 2

[15] L. Huang, C. Chen, W. Li, and Q. Du, "Remote sensing image scene classification using multi-scale completed local binary patterns and fisher vectors," *Remote Sensing*, vol. 8, no. 6, p. 483, 2016. 2

[16] G. Cheng, J. Han, L. Guo, Z. Liu, S. Bu, and J. Ren, "Effective and efficient midlevel visual elements-oriented land-use classification using vhr remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 8, pp. 4238–4249, 2015. 2

[17] G. Cheng, J. Han, L. Guo, and T. Liu, "Learning coarse-to-fine sparselets for efficient object detection and scene classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1173–1181. 2

[18] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*. ACM, 2010, pp. 270–279. 2, 7, 8, 9, 10

[19] L.-J. Zhao, P. Tang, and L.-Z. Huo, "Land-use scene classification using a concentric circle-structured multiscale bag-of-visual-words model," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 12, pp. 4620–4631, 2014. 2

[20] S. Chen and Y. Tian, "Pyramid of spatial relatons for scene-level land use classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 4, pp. 1947–1957, 2014. 2

[21] B. Zhao, Y. Zhong, L. Zhang, and B. Huang, "The fisher kernel coding framework for high spatial resolution scene classification," *Remote Sensing*, vol. 8, no. 2, p. 157, 2016. 2

[22] H. Wu, B. Liu, W. Su, W. Zhang, and J. Sun, "Hierarchical coding vectors for scene level land-use classification," *Remote Sensing*, vol. 8, no. 5, p. 436, 2016. 2

[23] F. Hu, G.-S. Xia, Z. Wang, X. Huang, L. Zhang, and H. Sun, "Unsupervised feature learning via spectral clustering of multidimensional patches for remotely sensed scene classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 5, 2015. 2

[24] Y. Yu, Z. Gong, C. Wang, and P. Zhong, "An unsupervised convolutional feature fusion network for deep representation of remote sensing images," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 1, pp. 23–27, 2017. 2

[25] X. Lu, X. Zheng, and Y. Yuan, "Remote sensing scene classification by unsupervised representation learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 9, pp. 5148–5157, 2017. 2

[26] J. Fan, T. Chen, and S. Lu, "Unsupervised feature learning for land-use scene recognition," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 4, pp. 2250–2261, 2017. 2

[27] I. T. Jolliffe, "Principal components in regression analysis," in *Principal component analysis*. Springer, New York, NY, 1986, pp. 129–155. 2

[28] X. Zheng, X. Sun, K. Fu, and H. Wang, "Automatic annotation of satellite images via multifeature joint sparse coding with spatial relation constraint," *IEEE Geoscience and Remote Sensing Letters*, vol. 10, no. 4, pp. 652–656, 2012. 2

[29] A. M. Cheriyadat, "Unsupervised feature learning for aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 1, pp. 439–451, 2014. 2

[30] K. Qi, Z. Xiaochun, W. Baiyan, and H. Wu, "Sparse coding-based correlaton model for land-use scene classification in high-resolution remote-sensing images," *Journal of Applied Remote Sensing*, vol. 10, no. 4, p. 042005, 2016. 2

[31] F. Zhang, B. Du, and L. Zhang, "Saliency-guided unsupervised feature learning for scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 4, pp. 2175–2184, 2015. 2

[32] E. Othman, Y. Bazi, N. Alajlan, H. Alhichri, and F. Melgani, "Using convolutional features and a sparse autoencoder for land-use scene classification," *International Journal of Remote Sensing*, vol. 37, no. 10, pp. 2149–2167, 2016. 2

[33] X. Ma, H. Wang, and J. Geng, "Spectral–spatial classification of hyperspectral image based on deep auto-encoder," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 9, no. 9, pp. 4073–4085, 2016. 2

[34] W. Li, H. Fu, L. Yu, P. Gong, D. Feng, C. Li, and N. Clinton, "Stacked autoencoder-based deep learning for remote-sensing image

[35] G. Cheng, Z. Li, X. Yao, L. Guo, and Z. Wei, "Remote sensing image scene classification using bag of convolutional features," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 10, pp. 1735–1739, 2017. 2

[36] G. Cheng, P. Zhou, and J. Han, "Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 12, pp. 7405–7415, 2016. 2

[37] F. Hu, G.-S. Xia, J. Hu, and L. Zhang, "Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery," *Remote Sensing*, vol. 7, no. 11, pp. 14 680–14 707, 2015. 2

[38] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," *arXiv preprint arXiv:1508.00092*, 2015. 2

[39] D. Marmanis, M. Datcu, T. Esch, and U. Stilla, "Deep learning earth observation classification using imagenet pretrained networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 1, pp. 105–109, 2015. 2

[40] G. Cheng, C. Ma, P. Zhou, X. Yao, and J. Han, "Scene classification of high resolution remote sensing images using convolutional neural networks," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2016, pp. 767–770. 2

[41] K. Nogueira, O. A. Penatti, and J. A. dos Santos, "Towards better exploiting convolutional neural networks for remote sensing scene classification," *Pattern Recognition*, vol. 61, pp. 539–556, 2017. 2

[42] J. Xie, N. He, L. Fang, and A. Plaza, "Scale-free convolutional neural network for remote sensing scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, 2019. 2

[43] E. Othman, Y. Bazi, F. Melgani, H. Alhichri, N. Alajlan, and M. Zuair, "Domain adaptation network for cross-scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 8, pp. 4441–4456, 2017. 2

[44] S. Chaib, H. Liu, Y. Gu, and H. Yao, "Deep feature fusion for vhr remote sensing scene classification," *IEEE Trans. Geosci. Remote Sens*, vol. 55, no. 8, pp. 4775–4784, 2017. 2

[45] W. Zhao and S. Du, "Scene classification using multi-scale deeply described visual words," *International Journal of Remote Sensing*, vol. 37, no. 17, pp. 4119–4131, 2016. 2

[46] E. Li, J. Xia, P. Du, C. Lin, and A. Samat, "Integrating multilayer features of convolutional neural networks for remote sensing scene classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 10, pp. 5653–5665, 2017. 2

[47] Z. Chen, S. Wang, X. Hou, and L. Shao, "Recurrent transformer network for remote sensing scene categorisation." in *British Machine Vision Conference*, 2018. 2, 9, 10

[48] G. Cheng, P. Zhou, and J. Han, "Rifd-cnn: Rotation-invariant and fisher discriminative convolutional neural networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2884–2893. 2

[49] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys, "Ti-pooling: transformation-invariant pooling for feature learning in convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 289–297. 2

[50] Y. Mukuta and T. Harada, "Invariant tensor feature coding," *arXiv preprint arXiv:1906.01857*, 2019. 3, 5

[51] J. Sokolic, R. Giryes, G. Sapiro, and M. Rodrigues, "Generalization error of invariant classifiers," in *Artificial Intelligence and Statistics*, 2017, pp. 1094–1103. 3, 5

[52] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1449–1457. 3, 6, 8, 9

[53] C. Ionescu, O. Vantzos, and C. Sminchisescu, "Matrix backpropagation for deep networks with structured layers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2965–2973. 3

[54] T.-Y. Lin and S. Maji, "Improved bilinear pooling with cnns," *arXiv preprint arXiv:1707.06772*, 2017. 3, 6

[55] D. Acharya, Z. Huang, D. Pani Paudel, and L. Van Gool, "Covariance pooling for facial expression recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 367–374. 3, 5

[56] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell, "Compact bilinear pooling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 317–326. 3, 6

[57] S. Kong and C. Fowlkes, "Low-rank bilinear pooling for fine-grained classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 365–374. 3, 6

[58] Y. Cui, F. Zhou, J. Wang, X. Liu, Y. Lin, and S. Belongie, "Kernel pooling for convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2921–2930. 3

[59] X. Wei, Y. Zhang, Y. Gong, J. Zhang, and N. Zheng, "Grassmann pooling as compact homogeneous bilinear pooling for fine-grained visual classification," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 355–370. 3, 11

[60] P. Li, J. Xie, Q. Wang, and Z. Gao, "Towards faster training of global covariance pooling networks by iterative matrix square root normalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 947–955. 3, 5, 6, 7

[61] P. Li, J. Xie, Q. Wang, and W. Zuo, "Is second-order information helpful for large-scale visual recognition?" in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2070–2078. 5, 6

[62] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 6, 8, 9, 11

[63] S. Fiori, "Learning by natural gradient on noncompact matrix-type pseudo-riemannian manifolds," *IEEE transactions on neural networks*, vol. 21, no. 5, pp. 841–852, 2010. 6

[64] Z. Huang and L. Van Gool, "A riemannian network for spd matrix learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 7

[65] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998. 7

[66] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009. 7

[67] Q. Wang, S. Liu, J. Chanussot, and X. Li, "Scene classification with recurrent attention of vhr remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 2, pp. 1155–1167, 2018. 7, 8, 9, 10

[68] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 8, 9, 10, 11

[69] G. Cheng, C. Yang, X. Yao, L. Guo, and J. Han, "When deep learning meets metric learning: Remote sensing image scene classification via learning discriminative cnns," *IEEE Transactions on Geoscience and Remote Sensing*, 2018. 8, 9, 10

[70] N. He, L. Fang, S. Li, A. Plaza, and J. Plaza, "Remote sensing scene classification using multilayer stacked covariance pooling," *IEEE Transactions on Geoscience and Remote Sensing*, no. 99, pp. 1–12, 2018. 8, 9

[71] Y. Yu and F. Liu, "A two-stream deep fusion framework for high-resolution aerial scene classification," *Computational intelligence and neuroscience*, vol. 2018, 2018. 9

[72] D. Zeng, S. Chen, B. Chen, and S. Li, "Improving remote sensing scene classification by integrating global-context and local-object features," *Remote Sensing*, vol. 10, no. 5, p. 734, 2018. 8, 9

[73] W. Zhang, P. Tang, and L. Zhao, "Remote sensing image scene classification using cnn-capsnet," *Remote Sensing*, vol. 11, no. 5, p. 494, 2019. 8, 9

[74] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 618–626. 11