

# Exploiting representations from pre-trained convolutional neural networks for high-resolution remote sensing image retrieval

Yun Ge<sup>1,2</sup> · Shunliang Jiang<sup>1</sup> · Qingyong Xu<sup>1</sup> ·  
Changlong Jiang<sup>2</sup> · Famao Ye<sup>1</sup>

Received: 6 February 2017 / Revised: 13 September 2017 / Accepted: 13 October 2017 /  
Published online: 2 November 2017  
© Springer Science+Business Media, LLC 2017

**Abstract** With the increasing amount of high-resolution remote sensing images, it becomes more and more urgent to retrieve remote sensing images from large archives efficiently. The existing methods are mainly based on shallow features to retrieve images, while shallow features are easily affected by artificial intervention. Recently, convolutional neural networks (CNNs) are capable of learning feature representations automatically, and CNNs pre-trained on large-scale datasets are generic. This paper exploits representations from pre-trained CNNs for high-resolution remote sensing image retrieval. CNN representations from AlexNet, VGGM, VGG16, and GoogLeNet are first transferred for high-resolution remote sensing images, and then CNN features are extracted via two approaches. One is extracting the outputs of high-level layers directly and the other is aggregating the outputs of mid-level layers by means of average pooling with different pooling regions. Given the generalization and high dimensionality of the CNN features, feature combination and feature compression are also adopted to improve the feature representation. Experimental results demonstrate that aggregated features with pooling region smaller than the feature map size perform excellently, especially for VGG16 and GoogLeNet. Shallow feature makes a great contribution to enhance the retrieval precision when combined with CNN features, and compressed features reduce redundancy effectively. Compared with the state-of-the-art methods, the proposed feature extraction methods are very simple, and the features are able to improve retrieval performance significantly.

**Keywords** Pre-trained · CNN · Remote sensing · Image retrieval · Feature combination · Feature compression

---

✉ Famao Ye  
yefamao@ncu.edu.cn

Yun Ge  
geyun@nchu.edu.cn

<sup>1</sup> Information Engineering School, Nanchang University, Nanchang, China

<sup>2</sup> Software School, Nanchang Hangkong University, Nanchang, China

# 1 Introduction

As the development of remote sensing technology, a large quantity of high-resolution remote sensing images taken from satellites and aircrafts are readily available. How to analyze and make full use of the abundant spatial and contextual information provided by the remote sensing images has attracted more and more attentions in geoscience fields. In this case, to develop effective and efficient methods for retrieving remote sensing images from large archives becomes one of the challenging and emerging research topics.

Conventional remote sensing image retrieval systems retrieve images according to textual annotations and metadata stored in the archives [9]. The retrieval results rely on the availability and the quality of manual tags which are expensive to obtain and often ambiguous [10]. Then recent studies are shift to content-based remote sensing image retrieval (CBRSIR) systems, which retrieve images that are similar to a given query based on the visual content of images.

Feature extraction and similarity measure are two important components in CBRSIR systems. As a powerful feature representation can measure the image similarity more accuracy [42], thus most studies [1, 5, 8, 10, 12, 13, 15, 16, 21, 32, 37, 40, 42] focus on feature extraction to generate a numerical effective representation out of every image. Low-level features are extracted in the early stage of CBRSIR, such as color [5], texture [16], and shape [32]. However, it is very difficult for low-level features to represent the high-level semantic content of images [1, 10]. Then large “semantic gap”, i.e., the gap between low-level features and high-level semantics of an image, occurs when using low-level features to retrieve images. Various methods have been proposed to bridge the semantic gap which is the crucial challenge in CBRSIR. A direct approach is to employ relevance feedback schemes [10, 13] that take account of feedback to refine the search strategy and provide increasingly relevant images to a query. Another approach is to extract semantic features for image retrieval, such as region-level semantic mining [21] and scene semantic matching [37]. Compact features, which are aggregated based on local features [22], have also recently been adopted; such as bag of visual words (BoVW) [42], vector locally aggregated descriptors (VLAD) [28], and local structure learning (LSL) [12]. These previous methods have improved the retrieval performance to some extent, but the employed features are shallow features. Shallow features are largely hand-crafted, for example, color feature is extracted to represent an image artificially though such image has rich texture feature. Then the retrieval performance is affected by artificial intervention.

Convolutional neural networks (CNNs) capable of learning features automatically have recently become widely used in various image recognition tasks, including image classification, domain adaptation, scene recognition, image retrieval, and so on [2–4, 6, 7, 11, 14, 18, 20, 23, 25–27, 29, 31, 33–35, 43, 44]. CNN features are abstracted from low-level features, thus confining the semantic gap effectively. However, directly learning parameters of CNNs from only a few thousand training images is problematic [27]. Many works [2, 3, 11, 18, 25, 27] have demonstrated that representations learned with CNNs pre-trained on large-scale datasets (such as ImageNet) can be efficiently transferred to other visual recognition tasks with a limited amount of training data, even when the target dataset has different characteristics from the source dataset.

Fully-connected layers and convolutional layers of CNNs pre-trained on large-scale datasets are usually considered when repurposing to novel tasks on other datasets. The outputs of the fully-connected layers, which capture more high-level semantic features, can be used directly [11, 31] or by fine-tuning way [3, 27] to represent target images. The outputs of the convolutional layers, which can be interpreted as local features, are aggregated to

global features to represent target images [2, 23, 25, 35, 44]. The aggregation approaches mainly include encoding methods [25, 35] and pooling schemes [2, 23, 44]. Especially, Babenko and Lempitsky [2] have evaluated many aggregation approaches, and found that simple pooling schemes (such as sum-pooling) are better than encoding methods (such as fisher vector) for CNN features.

Representations of pre-trained CNNs are promising for high-resolution remote sensing images [6, 18, 29], given that they mostly describe landforms, which have strong low-level similarities with general-purpose optical images. In the high-resolution remote sensing image retrieval, the retrieved images often look alike but may be irrelevant to the query [38] because remote sensing images often represent large natural geographical scenes that contain abundant and complex visual contents [37]. Therefore, it is necessary to exploit representations from pre-trained CNNs for high-resolution remote sensing image retrieval. Many CNN features [17, 24, 45, 46] have been proposed to retrieve high-resolution remote sensing images very recently. The fully-connected features from a variety of CNNs are employed and the performance is superior to that of shallow features [24]. Fully-connected features and aggregated convolutional features are extracted from both the off-the-shelf CNNs and fine-tuned CNNs [17, 45, 46]. The aggregation methods used to compress last convolutional features in [17, 45, 46] are mainly BoVW, improved fisher kernel (IFK) [30], and VLAD [19], which belong to feature coding. Although max pooling, average pooling and IFK are compared in [45], IFK presents the best performance. However, such result is obtained based on the last convolutional layers from CNNs containing 8 weight layers. As regard to other deeper CNNs (such as VGG16 and GoogLeNet), the pooling scheme maybe a good choice according to [2]. Furthermore, various methods are utilized to improve the performance. Manual relevance feedback is used to improve the results [24], but the amount of manual work is also increased. A novel CNN architecture based on conventional convolution layers and a three-layer perceptron is proposed to learn low dimensional features in [46]. However, the performance of the novel CNN is very related to the training set for the limited number of the training images. Hu et al. [17] introduces multi-scale concatenation for last convolutional features and multi-patch pooling for fully-connected layers to improve the results. But multiple inputs are needed to re-feed to the CNNs, resulting in a relatively complex feature extraction process.

This paper focuses on simple but effective methods to extract the CNN features for remote sensing image retrieval. The parameters from four representative CNNs (AlexNet [20], VGGM [7], VGG16 [33], and GoogLeNet [34]), which are pre-trained on ImageNet, are firstly transferred to represent high-resolution remote sensing images. Then high-level (last pooling layer for GoogLeNet, fully-connected layers for other CNNs) features and mid-level (penultimate pooling layer and last two inception layers for GoogLeNet, last convolutional layers and last pooling layers for other CNNs) features are compared to retrieve high-resolution remote sensing images. Especially, the average pooling scheme with different pooling regions is proposed to aggregate mid-level features. The experiments show that the high-level features of VGGM, and the aggregated mid-level features of VGG16 and GoogLeNet present excellent results.

This paper also employs feature combination and feature compression to improve the retrieval performance, given the shortcomings of the CNN features. CNN features are too generalized and lack some characteristics of objects that are important for image retrieval; therefore, combinations of different CNN features and combinations of CNN features with a shallow feature are studied to improve feature representation. CNN features are high-dimensional, so principal component analysis (PCA) is chosen to compress them. The simulations show that the shallow feature makes a great contribution to improve the

performance when combined with CNN features, and most compressed high-level features improve the results.

The remainder of this paper is organized as follows. Section 2 introduces four representative CNNs and illustrates the proposed retrieval method. Feature combination and feature compression are also described in Section 2. Experiments are evaluated in Section 3, and the conclusions are presented in Section 4.

## 2 Methodology

This section analyzes various CNNs pre-trained on ImageNet. Representations from these CNNs are then transferred for high-resolution remote sensing image retrieval. Feature combination and feature compression are adopted to improve retrieval performance.

### 2.1 Four CNN architectures

Various CNNs have been proposed for computer vision tasks. Here we pay attention to four representative CNNs: AlexNet [20], VGGM [7], VGG16 [33], and GoogLeNet [34]. AlexNet is the base CNN, VGGM shows good generic ability, and VGG16 and GoogLeNet achieve high classification accuracy by adopting depth increasing and inception modules, respectively. The four pre-trained CNNs have their own characteristics and employ different mechanisms; we thus exploit all their representations for remote sensing image retrieval.

AlexNet consists of convolutional layers, pooling layers, and fully-connected layers. Rectification linear unit (relu) is used as an activation function to accelerate the ability to fit the training set. To reduce complex co-adaptations of neurons, dropout is adopted in the first two fully-connected layers. The output of the last fully-connected layer is fed to a 1000-way softmax, which produces a distribution over the 1000 class labels. AlexNet achieved a performance leap for image classification on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC 2012), then a variety of improvements have been proposed based on AlexNet [7, 33, 34].

Simonyan and Zisserman [33] found that significant improvement can be attained by pushing the convolutional network depth to 16–19 weight layers. The width of convolutional layers (the number of channels) is small, starting from 64 in the first layer and then doubled after each max-pooling layer, until it reaches 512. VGG16 containing 16 weight layers is chosen here, as it gives similar performance to 19 layers but is less computationally complex. Another architecture, VGGM, has presented excellent generic performance. Similar to the CNN architecture of Zeiler [43], VGGM decreases stride and uses the smaller receptive field of the first convolutional layer. The main difference between VGGM and the CNN of Zeiler is that the former uses fewer filters in the fourth convolutional layer to achieve higher classification accuracy.

GoogLeNet, which ranked first in ILSVRC 2014, is a 22-layer deep convolutional network that contains nine “inception modules” stacked on top of each other. Each inception module is restricted to filter sizes of  $1 \times 1$ ,  $3 \times 3$ , and  $5 \times 5$  to avoid patch-alignment issues. To reduce computational complexity,  $1 \times 1$  convolutions are used first for dimensionality reduction before the expensive  $3 \times 3$  and  $5 \times 5$  convolutions. GoogLeNet removes fully-connected layers and uses the linear layer to adapt to other label sets.

Table 1 lists details of the four CNN architectures. For simplicity, relu and softmax layers are not shown in the table. AlexNet takes RGB images of  $227 \times 227$  pixels as input, and the other CNNs require  $224 \times 224$  RGB images. The output sizes of the convolutional layers

**Table 1** Architectures of different CNNs

AlexNet	VGGM	VGG16	GoogLeNet
input 227×227×3	input 224×224×3	input 224×224×3	input 224×224×3
conv1 55×55×96	conv1 109×109×96	conv1-1 224×224×64 conv1-2 224×224×64	conv1 112×112×64
pool1 27×27×96	pool1 54×54×96	pool1 112×112×64	pool1 56×56×64
conv2 27×27×256	conv2 26×26×256	conv2-1 112×112×128 conv2-2 112×112×128	conv2 56×56×192
pool2 13×13×256	pool2 13×13×256	pool2 56×56×128	pool2 28×28×192
conv3 13×13×384	conv3 13×13×512	conv3-1 56×56×256 conv3-2 56×56×256 conv3-3 56×56×256	inception3a 28×28×256 inception3b 28×28×480
conv4 13×13×384	conv4 13×13×512	pool3 28×28×256 conv4-1 28×28×512 conv4-2 28×28×512 conv4-3 28×28×512	pool3 14×14×480 inception4a 14×14×512 inception4b 14×14×512 inception4c 14×14×512 inception4d 14×14×528 inception4e 14×14×832
conv5 13×13×256	conv5 13×13×512	pool4 14×14×512 conv5-1 14×14×512 conv5-2 14×14×512 conv5-3 14×14×512	pool4 7×7×832 inception5a 7×7×832 inception5b 7×7×1024
pool5 6×6×256	pool5 6×6×512	pool5 7×7×512	pool5 1×1×1024
fc6 1×1×4096	fc6 1×1×4096	fc6 1×1×4096	dropout 1×1×1024
fc7 1×1×4096	fc7 1×1×4096	fc7 1×1×4096	linear 1×1×1000
fc8 1×1×1000	fc8 1×1×1000	fc8 1×1×1000	

(conv for short), pooling layers (pool for short), fully-connected layer (fc for short), and inception layer (inception for short) are given in the table.

## 2.2 Transferring CNN representations for high-resolution remote sensing image retrieval

For pre-trained CNNs to be generally applicable, the parameters of their internal layers are transferred for high-resolution remote sensing images. The outputs of different layers influence the retrieval performance, making suitable layer selection a crucial problem during retrieval.

Low-level features from early layers do not describe an image well, mid-level layer features describe the local characteristics of an image, and high-level layer features represent the global semantic information of an image [43]. Therefore, the high-level features and mid-level features are extracted in this paper, and feature extraction methods of different layers are described as follows:

### (1) Features extracted from high-level layers

High-level layers refer to fc6 and fc7 for AlexNet, VGGM, and VGG16; and high-level layer means pool5 for GoogLeNet. The outputs of high-level layers tend to represent the semantic information of an image, thus the features extracted from high-level layers are performed following the standard pipeline of previous works [3, 11], which is denoted as direct

extraction method. A resized high-resolution image is input to pre-trained CNNs, and then the outputs of high-level layers are of dimension  $1 \times 1 \times 4096$  for AlexNet, VGGM, and VGG16 ( $1 \times 1 \times 1024$  for GoogLeNet), which are computed as 4096-dimensional vectors (1024-dimensional vector for GoogLeNet). The 4096-dimensional vectors (1024-dimensional vector for GoogLeNet) are extracted as CNN features from high-level layers directly.

(2) Features extracted from mid-level layers

Mid-level layers refer to conv5 (conv5-3 for VGG16) and pool5 for AlexNet, VGGM, and VGG16, and mid-level layers are pool4, inception5a, inception5b for GoogLeNet. For convenience, conv5 means conv5-3 for VGG16 in the following sections. The outputs of mid-level layers describe local characteristics very well, so aggregation methods are adopted to aggregate mid-level outputs to global features. The simple pooling schemes perform better than coding methods when aggregating CNN features to compact features for image retrieval in [2]. Here, we propose to use average pooling to aggregate the outputs of mid-level layers to compact features, and different pooling regions are also studied in this paper.

Given a resized image  $I$ , the output of a mid-level layer  $l$  is  $s^l \times s^l \times c^l$ ,  $s^l \times s^l$  is the size of each feature map and  $c^l$  denotes the number of channels. The pooling region is  $m^l \times m^l$ , which is denoted as  $f^l$ . The pooling stride is  $t$ . Then the number of pooling regions is  $(s^l - m^l + t) \times (s^l - m^l + t)$ , which is denoted as  $r^l$ . The average pooling can be defined as

$$p^l(i) = \frac{1}{m^l \times m^l} \sum f^l(i), i = 1, 2, \dots, r^l, \tag{1}$$

In many papers, the pooling region is equal to the feature map [23, 44]. In order to obtain more information of the complex remote sensing images, the pooling region  $m^l \times m^l$  is not greater than the feature map in this paper. Our simulations also show that the average pooling achieves good performance when the  $m^l \times m^l$  is slightly smaller than the feature map size for most mid-level features.

The retrieval flow chart in Fig. 1 describes the retrieval procedure of AlexNet, VGGM, and VGG16. That of GoogLeNet is similar but uses the different outputs given above. A remote sensing (RS) image dataset is denoted as  $\mathbf{M}$ , and a query image is denoted as  $q$ . Each resized image ( $227 \times 227$  pixels for AlexNet or  $224 \times 224$  pixels for others) of  $\mathbf{M}$  is

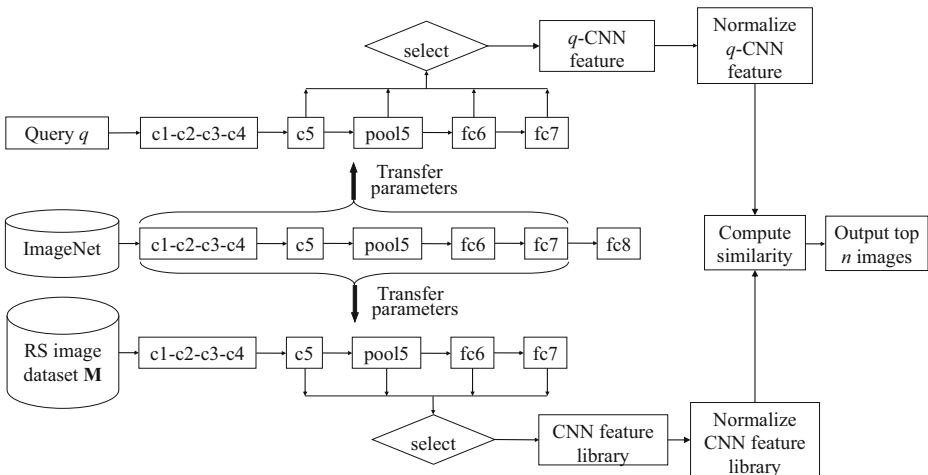


Fig. 1 Retrieval flow chart

input to the pre-trained CNN, and then a CNN feature library for  $\mathbf{M}$  is established by the sets of features extracted from a selected layer. A resized  $q$  is input to the pre-trained CNN, and  $q$ -CNN feature for  $q$  is the feature extracted from a selected layer. After normalizing the extracted CNN-features, the similarities between  $q$  and each image in  $\mathbf{M}$  are computed, and then the top  $n$  similar images are returned according to the sort of similarities.

### 2.3 Feature combination and feature compression

After the four representative pre-trained CNNs have been transferred for remote sensing image retrieval, feature combination and feature compression are studied to improve retrieval performance.

The retrieval performance of the proposed method is determined by the generality of the CNN features. The more general the features, the better the performance. Given that CNN features are abstracted from low-level features, they are often too generalized to include some important characteristics of objects. To improve feature representation, feature combination is studied from two aspects. On the one hand, to combine different CNN features by investigating three types of combinations. The first type is combining CNN features from the same CNN, but from different layers; the second type is combining CNN features from different CNNs, but from the same layer; the third type is combining CNN features not only from different CNNs, but also from different layers. On the other hand, CNN features (learned from the pre-trained CNNs) and shallow features (extracted based on hand-crafted features) are combined to achieve complementary effects.

The paper concentrates on the feature selection in the feature combination, then the simple combination strategy with weight allocation is adopted to combine different features. Given  $k$  normalized features  $\{F_1, F_2, \dots, F_k\}$  which are to be combined, query  $q$ , and a dataset  $\mathbf{M}$ . The similarity between  $q$  and  $\mathbf{M}$  is denoted as  $d_i$  that calculated according to  $F_i$ ,  $i = 1, \dots, k$ . Then the  $k$  features are combined by weighting the similarity as

$$d = \sum_{i=1}^k w_i \times d_i, \quad \text{where} \quad \sum_{i=1}^k w_i = 1, \quad (2)$$

$w_i$  denotes the weight of  $d_i$ . In other words,  $w_i$  reflects the weight of  $F_i$  because  $d_i$  is computed based on  $F_i$ . The calculated total similarity is denoted as  $d$ . Finally,  $d$  is needed to be normalized to make different features have the same range value in the similarity measure.

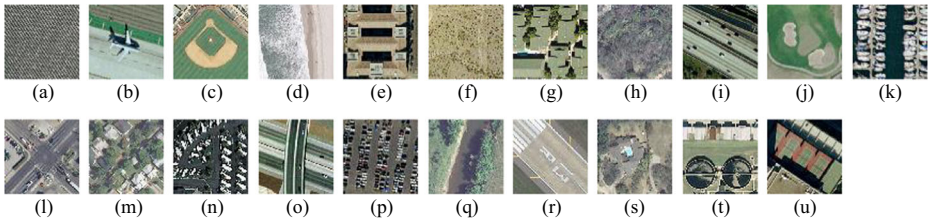
Feature compression is adopted to improve retrieval performance for the following two reasons. First, CNN feature dimensionality is often higher than that of shallow features, which leads to low retrieval efficiency. Second, redundancy is common when transferring CNN representations pre-trained on ImageNet for image retrieval on a small dataset. Therefore, this paper uses simple but efficient PCA methods to compress the CNN features to a number of different dimensions. These compressed features are then very helpful to improve the retrieval performance.

## 3 Experimental evaluation

The four pre-trained CNNs adopted here are developed using MatConvNet [36]. The following three high-resolution remote sensing image datasets are employed to do the experiments.

- (1) UC Merced Land Use dataset [41] (UC-Merced), which consists of 21 land-use categories selected from the United States Geological Survey (USGS) National Map. Each





**Fig. 2** Examples of the UC-Merced dataset: **a** agricultural; **b** airplane; **c** baseball diamond; **d** beach; **e** buildings; **f** chaparral; **g** dense residential; **h** forest; **i** freeway; **j** golf course; **k** harbor; **l** intersection; **m** medium density residential; **n** mobile home park; **o** overpass; **p** parking lot; **q** river; **r** runway; **s** sparse residential; **t** storage tanks; **u** tennis courts

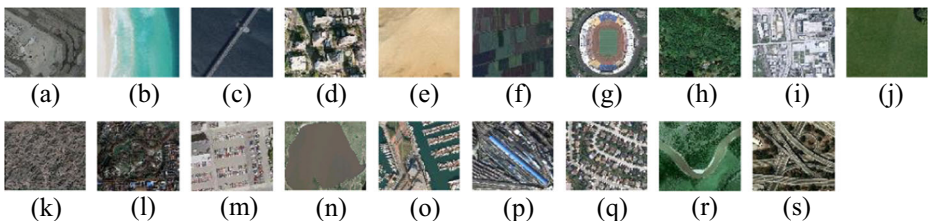
class consists of 100 images cropped to  $256 \times 256$  pixels, with a spatial resolution of one foot. Figure 2 shows one example of each category.

- (2) WHU-RS dataset [39], which contains 19 classes of satellite scene collected from Google Earth (Google Inc.). Each class has 50 images of  $600 \times 600$  pixels. An example of each class is shown in Fig. 3.
- (3) PatternNet dataset [47], which contains 38 classes collected from Google Earth imagery or via GoogleMap API for US cities very recently. There are a total of 800 images of size  $256 \times 256$  pixels in each class. Figure 4 presents some samples of PatternNet.

Similarity is obtained using L2-distance on L2-normalized CNN features. To evaluate comprehensively the retrieval performance, we compare the mean average precision (mAP) and average normalize modified retrieval rank (ANMRR) [42]. ANMRR was used extensively in the MPEG-7 standardization process; it considers both the number and order of the ground truth items that appear in the top retrievals. Good retrieval performance is indicated by a high mAP and a low ANMRR. Precision-recall curves for different features are also compared in the simulations.

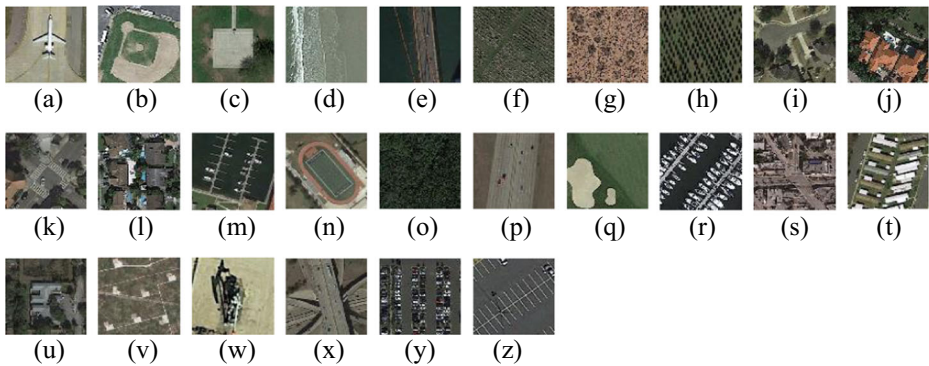
### 3.1 mAP

Tables 2 and 3 list mAP values for different features from high-level layers on the UC-Merced and the WHU-RS datasets, respectively. The results of fc6, relu6, fc7, and relu7 are compared for AlexNet, VGGM, and VGG16, and the results of pool5 are compared for GoogLeNet. The results of BoVW [42], which perform well in shallow features, are compared in both tables.



**Fig. 3** Examples of the WHU-RS dataset: **a** airport; **b** beach; **c** bridge; **d** commercial area; **e** desert; **f** farmland; **g** football field; **h** forest; **i** industrial area; **j** meadow; **k** mountain; **l** park; **m** parking lot; **n** pond; **o** port; **p** railway station; **q** residential area; **r** river; **s** viaduct





**Fig. 4** Examples of the WHU-RS dataset: **a** airplane; **b** baseball field; **c** basketball court; **d** beach; **e** bridge; **f** cemetery; **g** chaparral; **h** christmas tree farm; **i** closed road; **j** coastal mansion; **k** crosswalk; **l** dense residential; **m** ferry terminal; **n** football field; **o** forest; **p** freeway; **q** golf course; **r** harbor; **s** intersection; **t** mobile home park; **u** nursing home; **v** oil gas field; **w** oil well; **x** overpass; **y** parking lot; **z** parking space

For AlexNet, VGGM, and VGG16, mAP values from fc6 are better than that from relu6; and mAP values from relu7 are better than that from fc7. Generally speaking, the mAP values from fc6 are superior to that from fc7 except the value of AlexNet on the WHU-RS dataset. The results indicate that the features from fc7 and relu7 become too generalized for image retrieval. The features from fc6 are more discriminative than that from relu6 in the high-resolution remote sensing image retrieval.

Tables 2 and 3 show that all of the CNN features are much better than BoVW. GoogLeNet performs best on the UC-Merced dataset, while VGGM performs best on the WHU-RS dataset. AlexNet performs poor relatively on both datasets.

Tables 4 and 5 list mAP values for different features from mid-level layers on the UC-Merced and the WHU-RS datasets. The pooling stride is set to 1 to obtain more information of the remote sensing images. To overall compare the features, four mAP values for each kind of CNN features are shown in the tables. The first values are computed from the directly extracted features without pooling, the extraction approach of which is the same as that of high-level features. That is the outputs of mid-level layers are converted to feature vectors directly. Other three values are computed from the aggregated features using average pooling. We compute the values for the aggregated features with pooling region from 2×2 to the feature map size. Top two best values for each feature, and the values computed using the common average pooling method which has the pooling region equalling the feature map size, are selected to be displayed in the two tables. The four mAP values for VGGM from

**Table 2** mAP values for features from high-level layers on the UC-Merced dataset

Features	fc6	relu6	fc7	relu7	pool5
AlexNet	51.76	48.20	48.83	51.56	
VGGM	55.67	52.85	54.13	54.95	
VGG16	55.23	53.21	53.14	53.20	
GoogLeNet					56.60
BoVW	30.36				

**Table 3** mAP values for features from high-level layers on the WHU-RS dataset

Features	fc6	relu6	fc7	relu7	pool5
AlexNet	62.55	60.92	61.07	63.48	
VGGM	65.12	63.71	62.43	64.04	
VGG16	63.47	63.25	61.38	62.18	
GoogLeNet					62.31
BoVW	38.94				

pool5 in Table 4 are taken as an example to illustrate the results clearly. The first value is 50.12, which is computed from the directly extracted features; the last value is 51.56, which is computed from the aggregated feature with  $6 \times 6$  pooling region, and the pooling region is equal to the feature map size; the second and the third values are computed from the aggregated features with  $4 \times 4$  pooling region and  $5 \times 5$  pooling region respectively, which are the top two best values and smaller than the feature map size.

Many interesting results can be obtained from Tables 4 and 5. Firstly, the first values are the worst results for each kind of CNN features, which indicate that the aggregated features perform better than the directly extracted features. Secondly, most of the best aggregated features have pooling regions that are smaller than the feature map sizes. On the UC-Merced dataset, the pooling regions for most of the best aggregate features are slightly larger than or equal to half the size the feature maps. Such as the best aggregated feature for AlexNet from pool5 has  $3 \times 3$  pooling region, which is equal to half the size of the feature map, and the best aggregated feature for VGGM from pool5 has  $4 \times 4$  pooling region, which is slightly larger than half the size of the feature map. On the WHU-RS dataset, the best aggregated features have pooling regions which are equal to or slightly smaller than the feature map sizes. For example, the best aggregated feature for VGG16 from conv5 has  $12 \times 12$  pooling

**Table 4** mAP values for features from mid-level layers on the UC-Merced dataset

Features	conv5	relu5	pool5	pool4	incep5a	incep5b
AlexNet	29.17	38.05	45.24			
	44.16 $7 \times 7$	42.87 $7 \times 7$	47.50 $3 \times 3$			
	43.97 $8 \times 8$	42.85 $11 \times 11$	47.33 $4 \times 4$			
	41.77 $13 \times 13$	42.73 $13 \times 13$	46.44 $6 \times 6$			
VGGM	29.14	42.83	50.12			
	49.22 $8 \times 8$	50.35 $6 \times 6$	53.58 $4 \times 4$			
	49.25 $9 \times 9$	50.29 $7 \times 7$	53.16 $5 \times 5$			
	46.91 $13 \times 13$	49.68 $13 \times 13$	51.56 $6 \times 6$			
VGG16	33.30	50.68	53.36			
	55.56 $10 \times 10$	57.06 $8 \times 8$	59.18 $4 \times 4$			
	55.55 $11 \times 11$	57.07 $9 \times 9$	59.00 $5 \times 5$			
	53.41 $14 \times 14$	53.93 $14 \times 14$	55.32 $7 \times 7$			
GoogLeNet				56.52	55.30	53.28
				59.36 $4 \times 4$	57.29 $5 \times 5$	56.48 $5 \times 5$
				59.54 $5 \times 5$	57.20 $6 \times 6$	56.55 $6 \times 6$
				58.63 $7 \times 7$	56.92 $7 \times 7$	56.32 $7 \times 7$

**Table 5** mAP values for features from mid-level layers on the WHU-RS dataset

Features	conv5	relu5	pool5	pool4	incep5a	incep5b
AlexNet	38.91	49.99	55.77			
	60.70 9×9	56.31 11×11	58.81 4×4			
	60.82 10×10	56.57 12×12	59.48 5×5			
	59.75 13×13	56.60 13×13	59.79 6×6			
VGGM	40.96	53.76	59.71			
	66.07 10×10	62.21 9×9	65.09 4×4			
	66.20 11×11	62.33 10×10	65.50 5×5			
	65.56 13×13	61.17 13×13	65.06 6×6			
VGG16	39.15	56.19	58.99			
	67.50 12×12	63.65 12×12	66.20 5×5			
	67.42 13×13	63.62 13×13	65.53 6×6			
	67.06 14×14	63.63 14×14	65.53 7×7			
GoogLeNet				63.26	61.79	58.84
				67.48 5×5	64.46 5×5	62.67 5×5
				67.54 6×6	64.69 6×6	63.19 6×6
				67.10 7×7	64.77 7×7	63.52 7×7

region, which is slightly smaller than the feature map size; and the best aggregated feature for GoogLeNet from inception5a has 7×7 pooling region, which is equal to the feature map size. Thirdly, on the two datasets, the best layers from which the aggregated features extracted are different for the same CNN, except GoogLeNet. For AlexNet, VGGM, and VGG16, the best layers are pool5 on the UC-Merced dataset, while the best layers are conv5 on the WHU-RS dataset. The reason may be related to the different sizes of the images on the two datasets. The input images are resized from 256×256 on the UC-Merced dataset, and from 600×600 on the WHU-RS dataset. Then the resized images on the WHU-RS dataset lose more information than that on the UC-Merced dataset, so the best layers for WHU-RS dataset are lower than that for UC-Merced dataset to obtain more object characteristics. For GoogLeNet, the best layers are pool4 on both datasets, given that the architecture of GoogLeNet is different with that of other three networks, and the mid-level layers of GoogLeNet perform well.

The features from high-level layers are compared with the features from mid-level layers from Tables 2 and 4, and 3 and 5. The features extracted directly from mid-level layers perform worse than the features from high-level features, except from pool4 for GoogLeNet on the WHU-RS dataset. But the performances of the aggregated features from mid-level layers are improved at a higher degree. For VGG16 and GoogLeNet, most aggregated features perform better than the features from high-level layers, especially the worst aggregated features are superior to the best high-level features for GoogLeNet on both datasets, and for VGG16 on the WHU-RS dataset. For AlexNet and VGGM, most aggregated features are inferior to the high-level features, especially on the UC-Merced dataset. Therefore, aggregation method is very useful for VGG16 and GoogLeNet, because the two CNNs are very deep and the features from mid-level layers describe discriminative local information.

In the following simulations, we consider only two kinds of features for each CNN to evaluate the experiments. One is the feature with the best mAP value from high-level layer, which uses the acronym *h* as the suffix. On the UC-Merced dataset, AlexNet-h, VGGM-h, and VGG16-h denote the features from fc6; GoogLeNet-h denotes the feature

from pool5. On the WHU-RS dataset, AlexNet-h denotes the feature from relu7; VGGM-h and VGG16-h denote the features from fc6; GoogLeNet-h denotes the feature from pool5. BoVW are also compared with the features from high-level layers. The other is the feature with the best mAP value from mid-level layer, which adopts the acronym *m* as the suffix. On the UC-Merced dataset, AlexNet-m denotes the aggregated feature from pool5 with 3×3 pooling region; VGGM-m denotes the aggregated feature from pool5 with 4×4 pooling region; VGG16-m denotes the aggregated feature from pool5 with 4×4 pooling region; GoogLeNet-m denotes the aggregated feature from pool4 with 5×5 pooling region. On the WHU-RS dataset, AlexNet-m denotes the aggregated feature from conv5 with 10×10 pooling region; VGGM-m denotes the aggregated feature from conv5 with 11×11 pooling region; VGG16-m denotes the aggregated feature from conv5 with 12×12 pooling region; GoogLeNet-m denotes the aggregated feature from pool4 with 6×6 pooling region.

### 3.2 Per class mAP

Figure 5 compares mAP values from high-level layers for each class on the UC-Merced dataset. The CNN features are generally better than BoVW, though AlexNet-h performs a little worse than BoVW for harbor. GoogLeNet-h performs best for many classes, particularly airplane and tennis courts. VGG16-h achieves noticeably better results than the other features for beach, harbor, and overpass, but its poor performance for baseball diamond greatly reduces its average value. VGGM-h gives a stable performance for all classes, while AlexNet-h performs worse than the other three CNN features for most classes.

Figure 6 shows mAP values from mid-level layers for each class on the UC-Merced dataset. VGG16-m and GoogLeNet-m outperform AlexNet-m and VGGM-m, except for buildings and golf course. Thus the aggregated features of VGG16 and GoogLeNet, which are very deep CNNs, achieve good performance for most kinds of classes.

Figure 7 compares mAP values from high-level layers for each class on the WHU-RS dataset. BoVW performs well for airport and beach, especially for meadow. But for other classes, BoVW performs worse than other CNN features. Of the CNN features, VGGM-h is the best and GoogLeNet-h is the worst. The performances of AlexNet-h and VGG16-h appear to depend on the type of image.

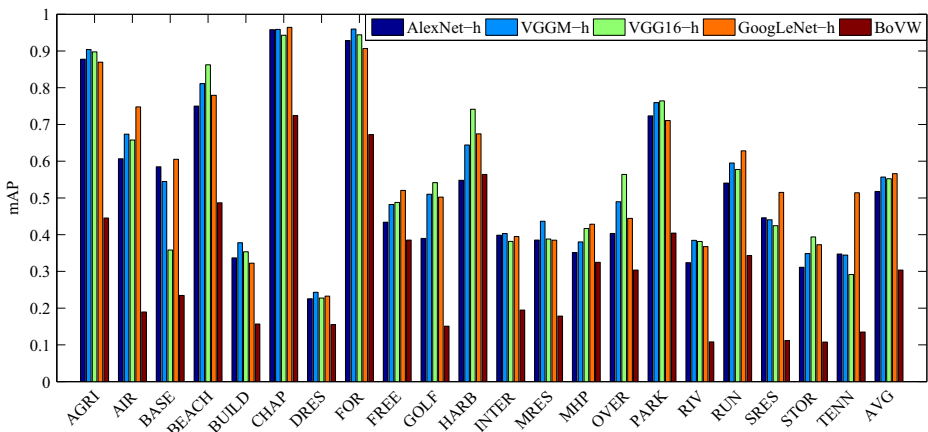


Fig. 5 Per class mAP values for high-level features on the UC-Merced dataset

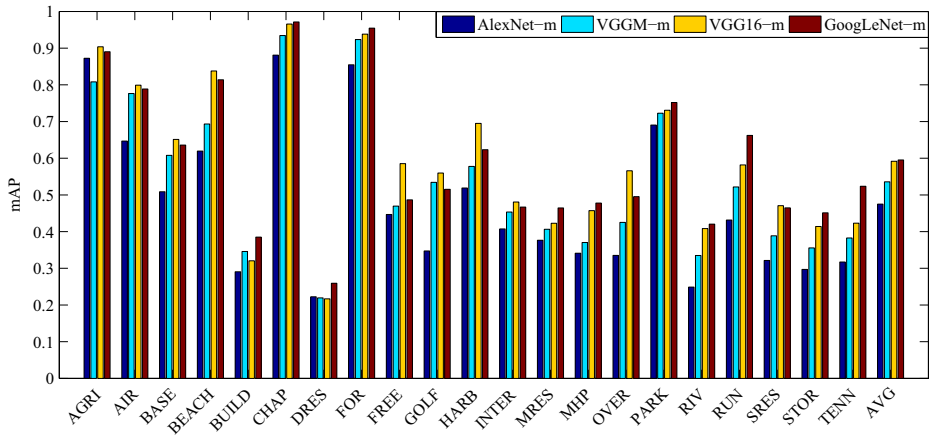


Fig. 6 Per class mAP values for mid-level features on the UC-Merced dataset

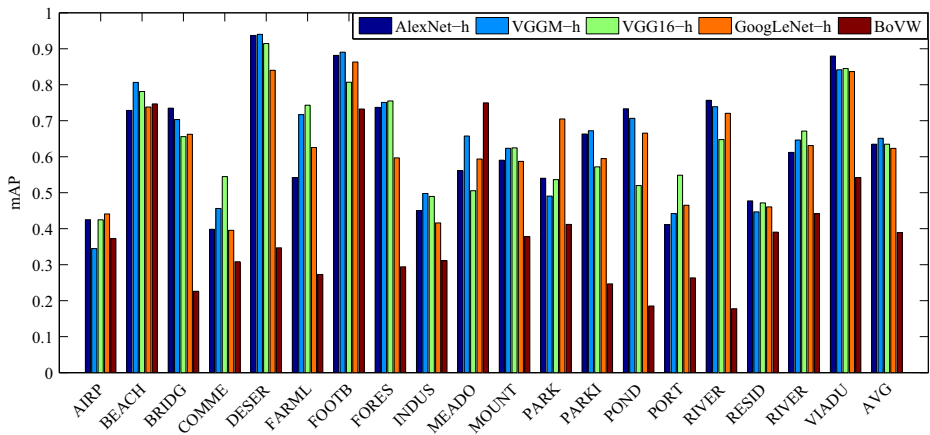


Fig. 7 Per class mAP values for high-level features on the WHU-RS dataset

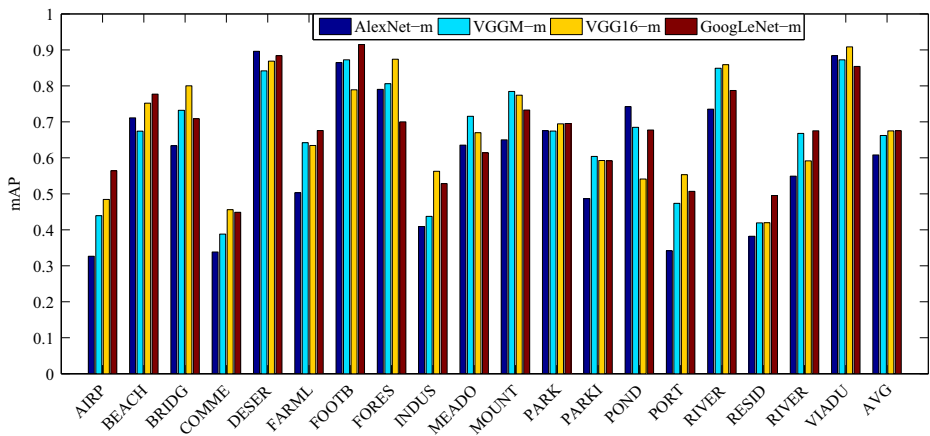


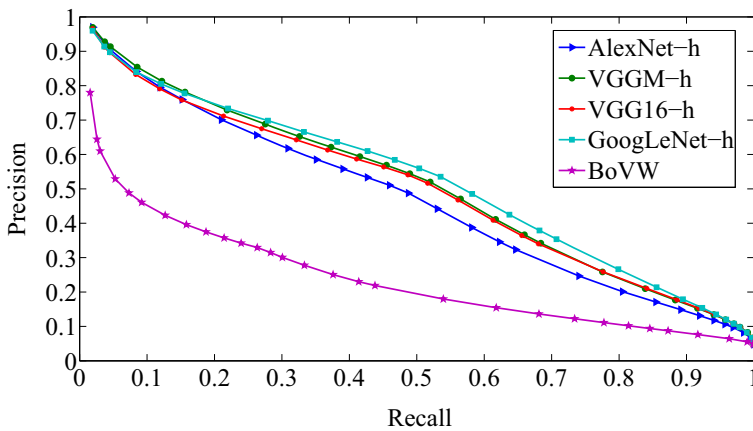
Fig. 8 Per class mAP values for mid-level features on the WHU-RS dataset

Figure 8 shows mAP values from mid-level layers for each class on the WHU-RS dataset. Similar to the results of Fig. 6, the aggregated features of VGG16 and GoogLeNet perform better for most kinds of classes than that of AlexNet and VGGM.

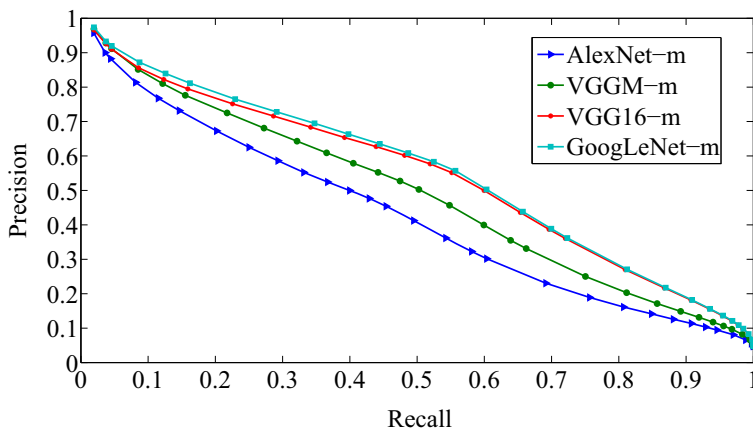
Overall, GoogLeNet-h presents the best results on the UC-Merced dataset, but performs worse on the WHU-RS dataset. VGGM-h performs stably on both datasets, and the average result of which is very close to that of GoogLeNet-h on the UC-Merced dataset. GoogLeNet-m and VGG16-m perform better than AlexNet-m and VGGM-m on both datasets. Therefore, we conclude that VGGM-h, GoogLeNet-m, and VGG16-m are more suitable for high-resolution remote sensing images.

### 3.3 Precision-recall curves

Figures 9 and 10 show precision-recall curves for the different features. Precision and recall are calculated for increasing numbers of retrieved images: from 2 to 2100 on the

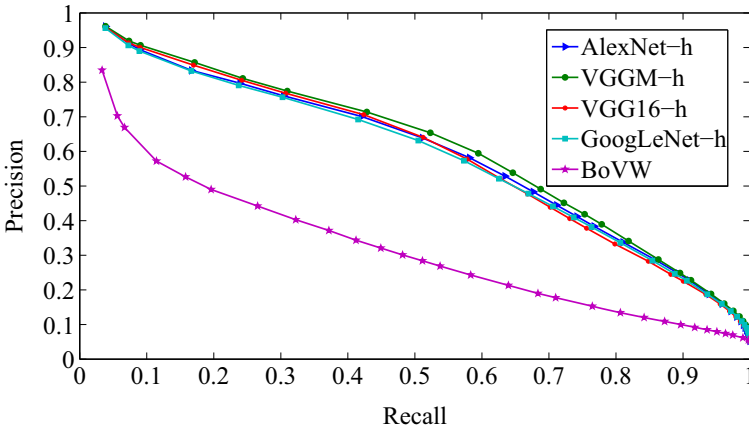


(a) high-level layer.

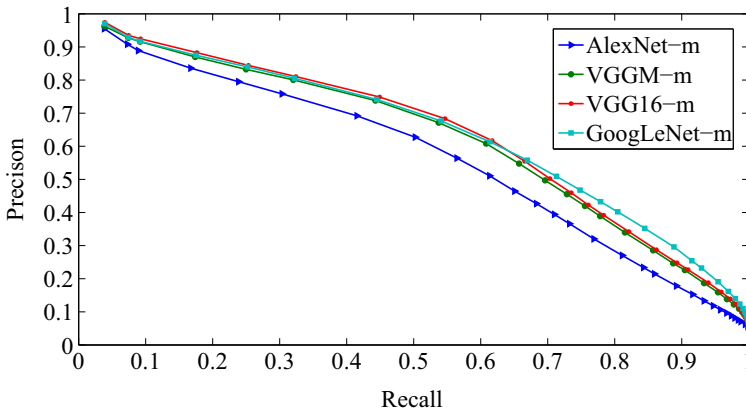


(b) mid-level layer.

**Fig. 9** Precision-recall curves for different features on the UC-Merced dataset



(a) high-level layer.



(b) mid-level layer.

**Fig. 10** Precision-recall curves for different features on the WHU-RS dataset

UC-Merced dataset, and from 2 to 950 on the WHU-RS dataset. On the UC-Merced dataset, BoVW shows a marked decline, especially once the number of returned images reaches 20, while the CNN features generally give more stable results. VGGM-h presents the best retrieval performance when the number of returned images is less than 20, but is surpassed by GoogLeNet-h as the number of returned images increases. GoogLeNet-m and VGG16-m show better results of these CNN features from mid-level layers, and when the number of returned images achieves 150, the result of VGG16-m is very close to that of GoogLeNet-m.

On the WHU-RS dataset, Fig. 10a shows that VGGM-h performs best, while BoVW is the worst. When the number of returned images is more than 40, the result of AlexNet-h becomes better and better. GoogLeNet-h presents worse result at the beginning, but GoogLeNet-h surpasses VGG16-h when the returned image number is more than 50. Figure 10b shows that AlexNet-m performs worst, and the result of VGGM-m is close to that of GoogLeNet-m and VGG16-m. When the returned image number is more than 40, the result of GoogLeNet-m becomes better than that of VGG16. Thus GoogleNet perform well when returned a large number of images.



### 3.4 ANMRR and dimensionality

Table 6 lists the results of ANMRR and dimensionality for the different features. As for shallow features, Aptoula improved remote sensing image retrieval performance with global morphological texture descriptors [1]. BoVW [42], VLAD [28], and LSL [12] are the aggregated features based on the 128-dimensional scale invariant feature transform (SIFT) descriptors. From Table 6, the ANMRR values of shallow features are worse than that of

**Table 6** ANMRR and dimensionality for different features

Features	Dimension		ANMRR	
	UC-Merced	WHU-RS	UC-Merced	WHU-RS
BoVW [42]	150	150	0.6010	0.5250
BoVW [42]	15000	15000	0.5910	0.4920
Global morphological texture [1]	62	–	0.5750	–
VLAD [28]	16384	–	0.4604	–
LSL [12]	2048	–	0.5556	–
VGGM-fc [24]	4096	4096	0.3780	0.4190
VGGM-fc+RF [24]	4096	4096	0.3160	0.3400
VGGM-fc [45]	4096	–	0.4580	–
VGGM-conv5+IFK [45]	102400	–	0.4473	–
VGGM(FT)-fc [45]	4096	–	0.2166	–
VGGM(FT)-conv5+IFK [45]	102400	–	0.3892	–
AlexNet-conv5+VLAD [46]	25600	25600	0.5510	0.4190
AlexNet-conv5+IFK [46]	51200	51200	0.5000	0.4170
VGGM-fc [46]	4096	4096	0.3780	0.3000
VGGM(FT)-fc [46]	4096	4096	0.3290	0.0220
VGGM-conv5+VLAD [46]	51200	51200	0.5310	0.3520
VGGM-conv5+IFK [46]	102400	102400	0.4580	0.3820
VGG16-fc [46]	4096	4096	0.3940	0.3240
VGG16-conv5+VLAD [46]	51200	51200	0.5330	0.3420
VGG16-conv5+IFK [46]	102400	102400	0.4070	0.3680
LDCNN[46]	30	30	0.4390	0.0190
GoogLeNet(FT)+BoVW [17]	1000	–	0.4230	–
GoogLeNet(FT)+MultiPatch [17]	1024	–	0.3140	–
GoogLeNet(FT)+BoVW+PCA [17]	32	–	0.3350	–
GoogLeNet(FT)+MultiPatch+PCA [17]	32	–	0.2850	–
AlexNet-h	4096	4096	0.4053	0.2937
AlexNet-m	4096	4096	0.4435	0.3181
VGGM-h	4096	4096	0.3699	0.2816
VGGM-m	4608	4608	0.3877	0.2735
VGG16-h	4096	4096	0.3748	0.3000
VGG16-m	8192	4608	0.3354	0.2660
GoogLeNet-h	1024	1024	0.3509	0.2999
GoogLeNet-m	7488	3328	0.3306	0.2568

CNN features, except VLAD [28]. Thus CNN features improve the retrieval performance significantly.

With respect to CNN features, the results of state-of-the-art methods and our methods are listed. We firstly compare the features from high-level layers. VGGM-fc [24], VGGM-fc [45], VGGM-fc [46], and VGGM-h are the outputs from fully-connected layers. However, the ANMRR values of these features are different. VGGM-fc [24] and VGGM-fc [46] select normalized feature from the fc7. VGGM-fc [45] chooses fc7 feature without normalized. While VGGM-h is the L2-normalized features from fc6. Thus these specific details, such as the choice of layer and whether to use the normalization, affect the retrieval results.

Secondly, the features from mid-level layers are compared. Other papers adopt coding methods to aggregate convolutional features. Such as IFK and VLAD are applied on the convolutional features of AlexNet, VGGM, and VGG16 (AlexNet-conv5+VLAD [46], AlexNet-conv5+IFK [46], VGGM-conv5+VLAD [46], VGGM-conv5+IFK [45, 46], VGG16-conv5+VLAD [46] and VGG16-conv5+IFK [46]), BoVW is utilized to aggregate the convolutional features from fine-tuned (FT for short) GoogLeNet (GoogLeNet(FT)+BoVW[17]). However, these aggregated convolutional features perform worse than fully-connected features, except for VGGM-conv5+IFK [45]. Our paper employs the average pooling to aggregate the mid-level features. The pooling method is not only simpler than the coding approaches, but also achieves better results. For example, VGG16-m and GoogLeNet-m are superior to VGG16-h and GoogLeNet-h.

Finally, some approaches utilized to improve the performance are analyzed. The most important method is the fine-tuning process, and the training set has a significant impact on the results. For example, VGGM(FT)-fc [45, 46] is the fully-connected features from fine-tuned VGGM, and the improvement of VGGM(FT)-fc [45] is greater than that of VGGM(FT)-fc [46]. Because VGGM is fine-tuned by using the random training images from UC-Merced datasets in [45], while another training set independent of UC-Merced is employed to fine-tune VGGM in [46]. In the paper [46], even the same training set is utilized to fine-tune VGGM, the result of VGGM(FT)-fc [46] on the WHU-RS dataset is obvious better than that on the UC-Merced dataset. Particularly, due to the limited training set, the proposed low dimensional CNN (LDCNN) [46] perform worse than that extracted from pre-trained CNN directly (VGGM-fc [46]). In addition to fine-tuning, there exist other improved methods containing manual relevance feedback (VGGM-fc+RF) [24] and multi-patch mean pooling (GoogLeNet(FT)+MultiPatch) [17]. Manual relevance feedback increases the amount of manual work. Multi-patch mean pooling needs multiple inputs to re-feed to the CNN, resulting in a relatively complex feature extraction process.

Therefore, our proposed CNN features are simple and effective, especially the simple average pooling method achieves better results than the coding methods used in other papers. When other CNN features employ fine-tuning process, manual relevance feedback, or multi-patch mean pooling, most of the results are improved to a large degree. Even so, vgg16-m, GoogLeNet-m, and GoogLeNet-h perform better than some of the fine-tuned features, which are VGGM-conv5+IFK [45] and GoogLeNet(FT)+BoVW [17]. This paper also adopts feature combination and feature compression to improve the retrieval performance, the simulation results of which can be seen in the next two subsections.

### 3.5 Feature combination

It is easy for CNN features to be too generalized, and so miss some characteristics of objects. In such case, multiple features can be combined to improve the retrieval performance. Given the poor performance of AlexNet, we consider combinations of VGGM, VGG16,

**Table 7** ANMRR for multiple feature combinations on the UC-Merced dataset

Features	Weight	ANMRR
(M-h, M-m)	(0.6, 0.4)	0.3644
	(0.7, 0.3)	0.3645
	(0.8, 0.2)	0.3654
(M-h, BoVW)	(0.9, 0.1)	0.3531
	(0.8, 0.2)	0.3620
	(1.0, 0.0)	0.3699
(M-m, BoVW)	(0.9, 0.1)	0.3692
	(0.8, 0.2)	0.3791
	(1.0, 0.0)	0.3877
(16-h, 16-m)	(0.2, 0.8)	0.3329
	(0.1, 0.9)	0.3331
	(0.3, 0.7)	0.3344
(16-h, BoVW)	(0.9, 0.1)	0.3569
	(0.8, 0.2)	0.3638
	(1.0, 0.0)	0.3748
(16-m, BoVW)	(0.9, 0.1)	0.3237
	(1.0, 0.0)	0.3354
	(0.8, 0.2)	0.3465
(G-h, G-m)	(0.4, 0.6)	0.3222
	(0.3, 0.7)	0.3226
	(0.5, 0.5)	0.3231
(G-h, BoVW)	(0.9, 0.1)	0.3282
	(0.8, 0.2)	0.3497
	(1.0, 0.0)	0.3509
(G-m, BoVW)	(0.9, 0.1)	0.3162
	(1.0, 0.0)	0.3306
	(0.8, 0.2)	0.3362
(M-h, 16-h, g-h)	(0.2, 0.2, 0.6)	0.3138
	(0.3, 0.2, 0.5)	0.3142
	(0.2, 0.3, 0.5)	0.3146
(M-h, 16-h, g-h, BoVW)	(0.2, 0.2, 0.5, 0.1)	0.3042
	(0.3, 0.1, 0.5, 0.1)	0.3055
	(0.1, 0.3, 0.5, 0.1)	0.3059
(M-m, 16-m, g-m)	(0.0, 0.5, 0.5)	0.3035
	(0.1, 0.4, 0.5)	0.3037
	(0.1, 0.5, 0.4)	0.3041
(M-m, 16-m, g-m, BoVW)	(0.0, 0.4, 0.5, 0.1)	0.2964
	(0.1, 0.4, 0.4, 0.1)	0.2971
	(0.0, 0.5, 0.4, 0.1)	0.2972
(M-h, M-m, 16-h, 16-m, G-h, G-m)	(0.1, 0.0, 0.0, 0.4, 0.3, 0.2)	0.2960
	(0.1, 0.0, 0.0, 0.4, 0.4, 0.1)	0.2966
	(0.1, 0.0, 0.0, 0.4, 0.2, 0.3)	0.2969
(M-h, M-m, 16-h, 16-m, G-h, G-m, BoVW)	(0.1, 0.0, 0.0, 0.4, 0.3, 0.1, 0.1)	0.2915
	(0.1, 0.0, 0.0, 0.3, 0.4, 0.1, 0.1)	0.2916
	(0.1, 0.0, 0.0, 0.4, 0.4, 0.0, 0.1)	0.2926

GoogLeNet, and BoVW. Tables 7 and 8 list the results on the UC-Merced and the WHU-RS datasets respectively. In the two tables, M-h, M-m, 16-h, 16-m, G-h, and G-m are short for VGGM-h, VGGM-m, VGG16-h, VGG16-m, GoogLeNet-h, and GoogLeNet-m respectively. In order to obtain good combination performance, each weight is carried out from  $\{0.0, 0.1, 0.2, \dots, 1.0\}$ . The weights list in the tables are the top three best combination for the groups of features.

Besides the combination of CNN features with BoVW, three types of CNN feature combinations are compared in Tables 7 and 8. The first type is combining CNN features from the same CNN, but from different layers. The results are included in the first three lines. For example, the second line shows the results of the groups of two features containing 16-h, 16-m, and BoVW. The combination of 16-m and 16-h are the CNN features from VGG16, but from high-level layer and mid-level layer separately. The second type combining CNN features from different CNNs, but from the same layer. Here, the same layer mainly refers to the high-level layer or the mid-level layer. The results are included in the fourth line. For instance, the combination of M-h, 16-h, and G-h are the high-level features from VGGM, VGG16, and GoogLeNet respectively. The third type is combining CNN features not only from different CNNs, but also from different layers, which is shown in the fifth line.

From Tables 7 and 8, the third type of feature combination performs best, then is the second type, and the first type performs worst. Accordingly, the third type combines the maximum number of features, and the first type just combines two features.

Another interesting conclusion can be found out from the two tables, that is BoVW plays an very important role in the feature combination with a small portion. As shown in the above subsections, BoVW performs worse when comparing with single CNN feature. But when the combined features containing BoVW, the results are improved at a higher degree, especially on the WHU-RS dataset. However, compared with other combination groups, the weight has a greater impact on the results of the combinations of a CNN feature and BoVW. For example, on the WHU-RS dataset, when the weight of BoVW is less than or equal to 0.2, the results of the combination of 16-h and BoVW are superior to that of the combination of M-h, M-m, 16-h, 16-m, G-h, and G-m, which is the best combination among the combinations of CNN features. But when the weight of BoVW is larger than 0.2, the combination of 16-h and BoVW performs worse. On the UC-Merced dataset, the top three best weight schemes are (0.9, 0.1), (0.8, 0.2), and (1.0, 0.0), which indicate that when the weight of BoVW is larger than 0.2, the combination of a CNN feature and BoVW is inferior to the single CNN feature.

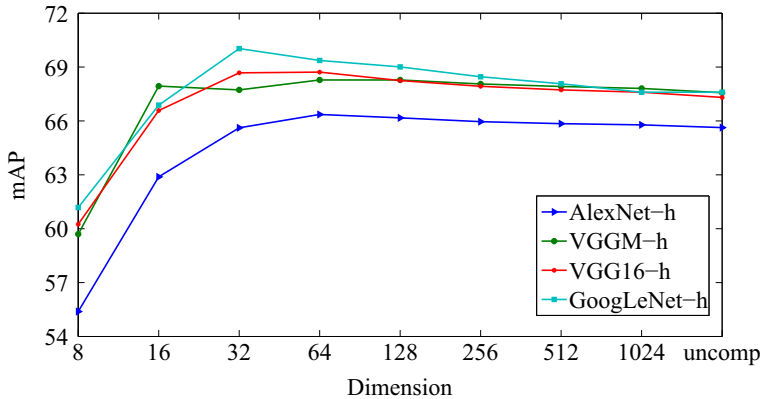
Therefore, feature combination is able to improve the feature representation, and the combination of hand-crafted features (such as BoVW) and CNN features should be a good choice.

### 3.6 Feature compression

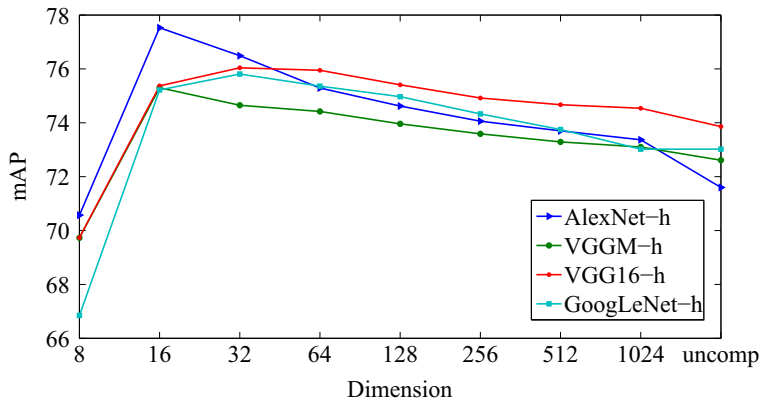
We randomly select 80% of the images from each category as the training set (1680 images for UC-Merced and 760 images for WHU-RS), and the rest are treated as testing set (420 images for UC-Merced and 190 images for WHU-RS). The projection matrices of PCA are learned on the training sets. Then the high dimensionality of the CNN features is compressed using PCA to various amounts between 8 and 1024 dimensions, as shown in Fig. 11 (which also include mAP values of the uncompressed features). When the PCA is employed on aggregated features from mid-level layers, which have been compressed using average

**Table 8** ANMRR for multiple feature combinations on the WHU-RS dataset

Features	Weight	ANMRR
(M-h, M-m)	(0.3, 0.7)	0.2427
	(0.2, 0.8)	0.2442
	(0.4, 0.6)	0.2451
(M-h, BoVW)	(0.8, 0.2)	0.1875
	(0.9, 0.1)	0.1914
	(0.7, 0.3)	0.2271
(M-m, BoVW)	(0.9, 0.1)	0.1878
	(0.8, 0.2)	0.2558
	(1.0, 0.0)	0.2735
(16-h, 16-m)	(0.2, 0.8)	0.2507
	(0.3, 0.7)	0.2532
	(0.1, 0.9)	0.2537
(16-h, BoVW)	(0.8, 0.2)	0.1967
	(0.9, 0.1)	0.2040
	(0.7, 0.3)	0.2345
(16-m, BoVW)	(0.9, 0.1)	0.1782
	(0.8, 0.2)	0.2514
	(1.0, 0.0)	0.2660
(G-h, G-m)	(0.2, 0.8)	0.2533
	(0.3, 0.7)	0.2536
	(0.1, 0.9)	0.2545
(G-h, BoVW)	(0.9, 0.1)	0.1998
	(0.8, 0.2)	0.2165
	(0.7, 0.3)	0.2717
(G-m, BoVW)	(0.9, 0.1)	0.1717
	(0.8, 0.2)	0.1822
	(0.7, 0.3)	0.2322
(M-h, 16-h, g-h)	(0.3, 0.3, 0.4)	0.2374
	(0.4, 0.2, 0.4)	0.2374
	(0.3, 0.2, 0.5)	0.2375
(M-h, 16-h, g-h, BoVW)	(0.3, 0.2, 0.4, 0.1)	0.1541
	(0.2, 0.3, 0.4, 0.1)	0.1552
	(0.4, 0.1, 0.4, 0.1)	0.1559
(M-m, 16-m, g-m)	(0.4, 0.4, 0.2)	0.2131
	(0.4, 0.3, 0.3)	0.2132
	(0.3, 0.4, 0.3)	0.2136
(M-m, 16-m, g-m, BoVW)	(0.2, 0.3, 0.4, 0.1)	0.1436
	(0.3, 0.2, 0.4, 0.1)	0.1437
	(0.1, 0.5, 0.3, 0.1)	0.1444
(M-h, M-m, 16-h, 16-m, G-h, G-m)	(0.0, 0.4, 0.1, 0.3, 0.1, 0.1)	0.2090
	(0.0, 0.5, 0.1, 0.2, 0.1, 0.1)	0.2091
	(0.1, 0.3, 0.0, 0.4, 0.1, 0.1)	0.2096
(M-h, M-m, 16-h, 16-m, G-h, G-m, BoVW)	(0.1, 0.2, 0.1, 0.3, 0.1, 0.1, 0.1)	0.1380
	(0.1, 0.2, 0.1, 0.2, 0.2, 0.1, 0.1)	0.1382
	(0.1, 0.3, 0.1, 0.1, 0.2, 0.1, 0.1)	0.1386



(a) UC-Merced.



(b) WHU-RS.

**Fig. 11** mAP vs dimension for different features on two datasets

pooling, the retrieval performance has poor improvement. Thus, the features from high-level layers are just considered in this experiment.

On the UC-Merced dataset, the best compressed dimensionality of AlexNet-h and VGG16-h is 32. The result of 64-dimensional VGGM-h is very close to that of 128-dimensional. GoogLeNet-h performs best when the dimensionality is reduced to 32. On the WHU-RS dataset, AlexNet-h and VGGM-h achieve the best results when reduced to 16 dimensions; VGG16-h and GoogLeNet-h perform best when reduced to 32 dimensions.

Therefore, the compressed features generally outperform the uncompressed features, especially when reduced to 32 and 64 dimensions for UC-Merced dataset; 16 and 32 dimensions for WHU-RS dataset. This might owing to a lot of redundancies when transferring CNNs pre-trained on ImageNet for these datasets.

The series of experimental results show that the CNN features give better results than shallow features. VGGM-h performs stably for the two datasets, VGG16-m and GoogLeNet-m increase the retrieval results greatly. Overall, feature combination and feature compression are very effective approaches to improve the retrieval performance.

**Table 9** Performance for different features on the PatternNet dataset

Features	Dimension	ANMRR	mAP
BoVW [47]	128	0.6393	27.29
VLAD [47]	8192	0.5686	33.67
IFK [47]	16384	0.6016	30.93
VGGM-h	4096	0.3118	62.40
VGGM-m	512	0.3205	61.50
VGG16-h	4096	0.3296	60.30
VGG16-m	2048	0.3044	62.92
GoogLeNet-h	1024	0.2978	63.21
GoogLeNet-m	832	0.2784	65.98

### 3.7 Large-scale dataset

In order to further study the scalability of the proposed method, a large-scale high-resolution remote sensing image dataset, PatternNet, is adopted to do experiments. CNN features are selected from VGGM, VGG16, and GoogLeNet.

The results of different features are presented in Table 9. BoVW, VLAD, and IFK are three representative shallow features, which are constructed by aggregating the 128-dimensional SIFT descriptors. For the CNN features, VGGM-h and VGG16-h are the high-level features from fc6, GoogLeNet-h is the high-level feature from pool5; VGGM-m and VGG16-m are the aggregated mid-level features from pool5, and GoogLeNet-m is the aggregated mid-level features from pool4. The pooling regions of VGGM-m, VGG16-m and GoogLeNet-m are  $6 \times 6$ ,  $6 \times 6$ , and  $7 \times 7$  respectively. Table 9 shows that CNN features perform better than shallow features. For these CNN features, VGG16-m and GoogLeNet-m are better than VGG16-h and GoogLeNet-h respectively, and VGGM-m is worse than VGG-h. Thus, the aggregated mid-level features present good performance for VGG16 and GoogLeNet.

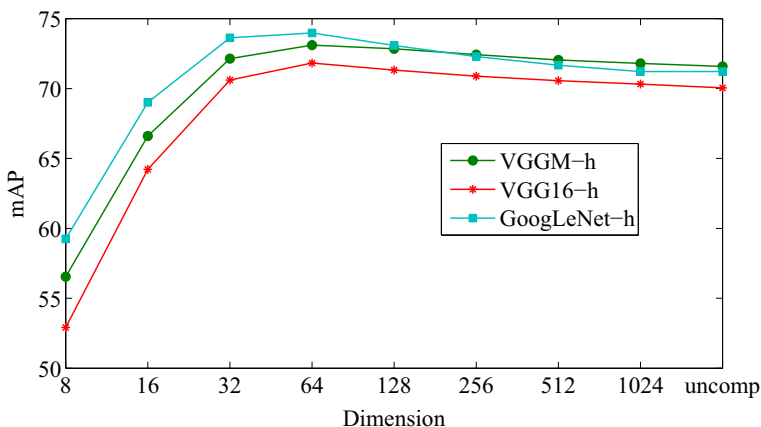
**Fig. 12** mAP vs dimension for different features on the PatternNet dataset



Figure 12 compares the results of compressed high-level features with various dimensions. Like the above feature compression experiment, we randomly select 80% of the images from each class of PatternNet as the training set and the rest are used for retrieval performance evaluation. Most compressed features perform better than the uncompressed features, especially the 32-dimensional and 64-dimensional features. Thus, feature compression is very useful to improve the CNN feature representation, even in the large-scale dataset.

## 4 Conclusions

We exploited representations of pre-trained CNNs (AlexNet, VGGM, VGG16, and GoogLeNet) based on image classification for high-resolution remote sensing image retrieval. Given the different characteristics between high-level layers and mid-level layers, direct extraction method and aggregation approach are adopted to extract high-level features and mid-level features respectively. Average pooling with different pooling regions is studied to aggregate the outputs of mid-level layers, and most aggregated features with pooling region smaller than the feature map size achieve excellent results. The series of experiments demonstrate that CNN features generally outperform shallow features except for a few image classes. High-level features for VGGM are very stable, and the average pooling is very useful to aggregate features, especially for VGG16 and GoogLeNet. Even on the large-scale image dataset, the CNN features present good performance. Compared to existing CNN features, the proposed feature extraction methods are not only simple, but also very effective. The presented average pooling method is suitable to aggregate CNN features.

We also demonstrated preliminary efforts to combine different features and to compress features to improve image retrieval according to the specific shortcomings of the CNN features; both greatly improved the retrieval performance. Many kinds of feature combinations are investigated, and particularly BoVW is found to play a great contribution to improve retrieval performance when combined with CNN features. Most compressed high-level features perform well when compressed to 32 dimensions. Adaptive feature fusion and further ways to aggregate features are planned in our future works.

**Acknowledgements** This work has been supported by National Natural Science Foundation of China [grant numbers 41261091, 61662044, 61663031, and 61762067].

## References

1. Aptoula E (2014) Remote sensing image retrieval with global morphological texture descriptors. *IEEE Trans Geosci Remote Sens* 52(5):3023–3034. <https://doi.org/10.1109/TGRS.2013.2268736>
2. Babenko A, Lempitsky V (2015) Aggregating local deep convolutional features for image retrieval. In: 15th IEEE international conference on computer vision, Santiago, Chile, pp 1269–1277. <https://doi.org/10.1109/ICCV.2015.150>
3. Babenko A, Slesarev A, Chigorin A, Lempitsky V (2014) Neural codes for image retrieval. In: 13th european conference on computer vision, Zurich, Switzerland, pp 584–599. [https://doi.org/10.1007/978-3-319-10590-1\\_38](https://doi.org/10.1007/978-3-319-10590-1_38)
4. Bai S, Li Z, Hou J (2017) Learning two-pathway convolutional neural networks for categorizing scene images. *Multimedia Tools and Applications* 76(15):16145–16162. <https://doi.org/10.1007/s11042-016-3900-6>

5. Bretschneider T, Cavet R, Kao O (2002) Retrieval of remotely sensed imagery using spectral information content. In: IEEE international geoscience and remote sensing symposium, Toronto, Canada, pp 2253–2255. <https://doi.org/10.1109/IGARSS.2002.1026510>
6. Castelluccio M, Poggi G, Sansone C, Verdoliva L (2015) Land use classification in remote sensing images by convolutional neural networks. *Acta Ecol Sin* 28(2):627–635
7. Chatfield K, Simonyan K, Vedaldi A, Zisserman A (2014) Return of the devil in the details: delving deep into convolutional networks. In: 25th british machine vision conference, Nottingham, England. <https://doi.org/10.5244/C.28.6>
8. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE computer society conference on computer vision and pattern recognition, San Diego, California, USA, pp 886–893. <https://doi.org/10.1109/CVPR.2005.177>
9. Datta R, Joshi D, Li J, Wang JZ (2008) Image retrieval: ideas, influences, and trends of the new age. *ACM Comput Surv* 40(2):5. <https://doi.org/10.1145/1348246.1348248>
10. Demir B, Bruzzone L (2015) A novel active learning method in relevance feedback for content-based remote sensing image retrieval. *IEEE Trans Geosci Remote Sens* 53(5):2323–2334. <https://doi.org/10.1109/TGRS.2014.2358804>
11. Donahue J, Jia Y, Vinyals O, Hoffman J, Zhang N, Tzeng E, Darrell T (2014) Decaf: a deep convolutional activation feature for generic visual recognition. In: 31st international conference on machine learning, Beijing, China, pp 647–655
12. Du Z, Li X, Lu X (2016) Local structure learning in high resolution remote sensing image retrieval. *Neurocomputing* 207:813–822. <https://doi.org/10.1016/j.neocom.2016.05.061>
13. Ferecatu M, Boujemaa N (2007) Interactive remote-sensing image retrieval using active relevance feedback. *IEEE Trans Geosci Remote Sens* 45(4):818–826. <https://doi.org/10.1109/TGRS.2007.892007>
14. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: 27th IEEE conference on computer vision and pattern recognition, Columbus, USA, pp 580–587. <https://doi.org/10.1109/CVPR.2014.81>
15. He Z, You X, Yuan Y (2009) Texture image retrieval based on non-tensor product wavelet filter banks. *Signal Process* 89(8):1501–1510. <https://doi.org/10.1016/j.sigpro.2009.01.021>
16. Hongyu Y, Bicheng L, Wen C (2004) Remote sensing imagery retrieval based-on gabor texture feature classification. In: 7th international conference on signal processing, pp 733–736. <https://doi.org/10.1109/ICOSP.2004.1452767>
17. Hu F, Tong X, Xia G, Zhang L (2016) Delving into deep representations for remote sensing image retrieval. In: 13th IEEE international conference on signal processing, Chengdu, China, pp 198–203. <https://doi.org/10.1109/ICSP.2016.7877823>
18. Hu F, Xia G, Hu J, Zhang L (2015) Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sens* 7:14680–14707. <https://doi.org/10.3390/rs71114680>
19. Jégou H, Douze M, Schmid C, Pérez P (2010) Aggregating local descriptors into a compact representation. In: IEEE conference on computer vision and pattern recognition, San Francisco, California, USA, pp 3304–3311. <https://doi.org/10.1109/CVPR.2010.5540039>
20. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: 26th conference on neural information processing systems, Nevada, US
21. Liu T, Zhang L, Li P, Lin H (2012) Remotely sensed image retrieval based on region-level semantic mining. *EURASIP Journal on Image and Video Processing* 4(1):1–11. <https://doi.org/10.1186/1687-5281-2012-4>
22. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110. <https://doi.org/10.1023/B:VISI.0000029664.99615.94>
23. Mousavian A, Zisserman J (2015) Deep convolutional features for image based retrieval and scene categorization. arXiv:1509.06033
24. Napoletano P (2016) Visual descriptors for content-based retrieval of remote sensing images. arXiv:1602.00970v1
25. Ng JY, Yang F, Davis LS (2015) Exploiting local features from deep networks for image. In: 28th IEEE conference on computer vision and pattern recognition workshops, Boston, MA, pp 53–61. <https://doi.org/10.1109/CVPRW.2015.7301272>

26. Ong EJ, Husain S, Bober M (2017) Siamese network of deep fisher-vector descriptors for image retrieval. arXiv:1702.00338
27. Oquab M, Bottou L, Laptev I, Sivic J (2014) Learning and transferring mid-level image representations using convolutional neural networks. In: 27th IEEE conference on computer vision and pattern recognition, Columbus, USA, pp 1717–1724. <https://doi.org/10.1109/CVPR.2014.222>
28. Ozkan S, Ates T, Tola E, Soysal M, Esen E (2014) Performance analysis of state-of-the-art representation methods for geographical image retrieval and categorization. *IEEE Geosci Remote Sens Lett* 11(11):1996–2000. <https://doi.org/10.1109/LGRS.2014.2316143>
29. Penatti OAB, Nogueira K, Santos JAD (2015) Do deep features generalize from everyday objects to remote sensing and aerial scenes domains. In: 28th IEEE conference on computer vision and pattern recognition, Boston, MA, pp 44–51. <https://doi.org/10.1109/CVPRW.2015.7301382>
30. Perronnin F, Sánchez J, Mensink T (2010) Improving the fisher kernel for large-scale image classification. In: 11th european conference on computer vision, Heraklion, Crete, Greece, pp 143–156. [https://doi.org/10.1007/978-3-642-15561-1\\_11](https://doi.org/10.1007/978-3-642-15561-1_11)
31. Razavian A, Azizpour H, Sullivan J, Carlsson S (2014) Cnn features off-the-shelf: an astounding baseline for recognition. In: 27th IEEE conference on computer vision and pattern recognition, Columbus, USA, pp 512–519. <https://doi.org/10.1109/CVPRW.2014.131>
32. Scott G, Klaric M, Davis C, Shyu CR (2011) Entropy-balanced bitmap tree for shape-based object retrieval from large-scale satellite imagery databases. *IEEE Trans Geosci Remote Sens* 49(5):1603–1616. <https://doi.org/10.1109/TGRS.2010.2088404>
33. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: 3rd international conference on learning representations, San Diego, California, USA
34. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V (2015) Going deeper with convolutions. In: 28th IEEE conference on computer vision and pattern recognition, Boston, MA, pp 1–9. <https://doi.org/10.1109/CVPR.2015.7298594>
35. Uricchio T, Bertini M, Seidenari L, Bimbo AD (2015) Fisher encoded convolutional bag-of-windows for efficient image retrieval and social image tagging. In: 15th IEEE international conference on computer vision workshop, Santiago, Chile, pp 1020–1026. <https://doi.org/10.1109/ICCVW.2015.134>
36. Vedaldi A, Lenc K (2015) Matconvnet: convolutional neural networks for MATLAB. In: 23rd ACM international conference on multimedia. Brisbane, Australia, pp 689–692. <https://doi.org/10.1145/2733373.2807412>
37. Wang M, Song T (2013) Remote sensing image retrieval by scene semantic matching. *IEEE Trans Geosci Remote Sens* 51(5):2874–2886. <https://doi.org/10.1109/TGRS.2012.2217397>
38. Wang Y, Zhang L, Tong X, Zhang L, Zhang Z, Liu H, Xing X, Mathiopoulos P (2016) A three-layered graph-based learning approach for remote sensing image retrieval. *IEEE Trans Geosci Remote Sens* 54(10):6020–6034. <https://doi.org/10.1109/TGRS.2016.2579648>
39. Xia G, Yang W, Delon J, Gousseau Y, Sun H, Maitre H (2010) Structural high-resolution satellite image indexing. In: ISPRS TC VII Symposium-100 years ISPRS 38, pp 298–303
40. Yan C, Zhang Y, Dai F, Zhang J, Li L, Dai Q (2014) Efficient parallel HEVC intra-prediction on many-core processor. *Electron Lett* 50(11):805–806. <https://doi.org/10.1049/el.2014.0611>
41. Yang Y, Newsam S (2010) Bag-of-visual-words and spatial extensions for land-use classification. In: 18th ACM SIGSPATIAL international conference on advances in geographic information systems, San Jose, California, pp 270–279
42. Yang Y, Newsam S (2013) Geographic image retrieval using local invariant features. *IEEE Trans Geosci Remote Sens* 51(2):818–832. <https://doi.org/10.1109/TGRS.2012.2205158>
43. Zeiler MD, Fergus R (2014) Visualizing and understanding convolutional networks. In: 13th european conference on computer vision, Zurich, Switzerland, pp 818–833. [https://doi.org/10.1007/978-3-319-10590-1\\_53](https://doi.org/10.1007/978-3-319-10590-1_53)
44. Zheng L, Zhao Y, Wang S, Wang J, Tian Q (2016) Good practice in cnn feature transfer. arXiv:1604.00133v1
45. Zhou W, Li C (2016) Deep feature representations for high-resolution remote-sensing imagery retrieval. arXiv:1610.03023
46. Zhou W, Newsam S, Li C, Shao Z (2017) Learning low dimensional convolutional neural networks for high-resolution remote sensing image retrieval. *Remote Sens* 9(5):489. <https://doi.org/10.3390/rs9050489>
47. Zhou W, Newsam S, Li C, Shao Z (2017) Patternnet: a benchmark dataset for performance evaluation of remote sensing image retrieval. arXiv:1706.03424



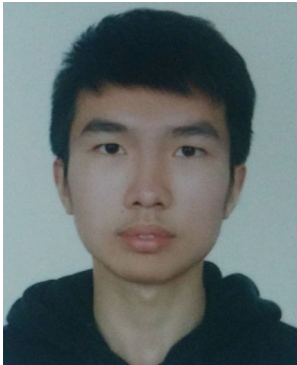
**Yun Ge** received the B.S. degree in Computer Science and Technology from HeFei University of Technology, Hefei, China, in 2005, and the M.S. degree in Computer Science from Central South University, Hunan, China, in 2009. She is currently pursuing the Ph.D. degree at the School of Information Engineering, Nanchang University, Nanchang, China. Her current research interests include machine learning and remote sensing image retrieval.



**Shunliang Jiang** is a professor in Computer Science & technology Department of Nanchang University, Nanchang, China. He received his bachelor degree from National University of Defense Technology at 1987, M.Sc. degree from Harbin Institute of Technology at 1990, PhD from Beijing University of Chemical Technology at 1996. His professional career included 5-year research experience in USA universities. His major interest focuses on the numerical methods with emphasis on the optimization and simulation of industry processing, and recent interests are in the areas of image processing, motion capture, and machine vision.



**Qingyong Xu** is currently pursuing the Ph.D. degree at the School of Information Engineering, Nanchang University and he has received the M.S. degree in Wuhan University, in 2006. His research interests include processing of image processing, pattern recognition and deep learning.



**Changlong Jiang** is currently pursuing the B.S. degree at the School of Software, Nanchang Hangkong University, Nanchang, Chian. His research interests include network security and image processing.



**Famao Ye** received the Ph.D. degree in cartography and geographical information system from Institute of Remote Sensing Application China Academy of Sciences, Beijing, China, in 2006. He is currently an associate professor with the School of Information Engineering, Nanchang University, Nanchang, China. His research interests include processing of remote sensing image and artificial intelligence.