remote sensing

MDPI

Article

High-Resolution Remote Sensing Image Retrieval Based on CNNs from a Dimensional Perspective

Zhifeng Xiao, Yang Long *, Deren Li, Chunshan Wei, Gefu Tang and Junyi Liu

State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China; xzf@whu.edu.cn (Z.X.); drli@whu.edu.cn (D.L.); cswei1874@163.com (C.W.); tanggefu@gmail.com (G.T.); liujunyi_ljy@163.com (J.L.)

* Correspondence: longyang@whu.edu.cn; Tel.: +86-189-0862-7409

Received: 26 May 2017; Accepted: 9 July 2017; Published: 14 July 2017

Abstract: Because of recent advances in Convolutional Neural Networks (CNNs), traditional CNNs have been employed to extract thousands of codes as feature representations for image retrieval. In this paper, we propose that more powerful features for high-resolution remote sensing image representations can be learned using only several tens of codes; this approach can improve the retrieval accuracy and decrease the time and storage requirements. To accomplish this goal, we first investigate the learning of a series of features with different dimensions using a few tens to thousands of codes via our improved CNN frameworks. Then, a Principal Component Analysis (PCA) is introduced to compress the high-dimensional remote sensing image feature codes learned by traditional CNNs. Comprehensive comparisons are conducted to evaluate the retrieval performance based on feature codes of different dimensions learned by the improved CNNs as well as the PCA compression. To further demonstrate the powerful ability of the low-dimensional feature representation learned by the improved CNN frameworks, a Feature Weighted Map (FWM), which can perform feature visualization and provides a better understanding of the nature of Deep Convolutional Neural Networks (DCNNs) frameworks, is explored. All the CNN models are trained from scratch using a large-scale and high-resolution remote sensing image archive, which will be published and made available to the public. The experimental results show that our method outperforms state-of-the-art CNN frameworks in terms of accuracy and storage.

Keywords: remote sensing image retrieval; Deep Compact Codes (DCC); Feature Weighted Map (FWM); CNN features; dimension reduction

1. Introduction

With the rapid development of Earth observation technology, remote imaging sensors with high spatial resolution have led to rapid increases in the volume of acquired remote sensing images. However, the effective management and retrieval of large scale remote sensing image databases represent considerable challenges that must be resolved. As a result, Content-Based High-Resolution Remote Sensing Imagery Retrieval (CB-HRRS-IR), which aims to search for and return the most relevant or similar images using a query image, has drawn increasing attention in recent years [1].

Currently, there are two essential modules that serve as solutions to CB-HRRS-IR: the feature representation module and the feature searching module [2]. Specifically, a feature vector is extracted to describe the visual content of an image in the feature representation module. Based on the extracted features, similarities between a query image and other images from the image database are calculated; then, the system returns the most similar images by ranking similarities. Both modules play important roles in an image retrieval system. Obviously, the length of the image features and the method of similarity measurement have a significant impact on the search efficiency, especially for enormous

image archives in which the extracted features can largely influence the retrieval performance because of the ability of the features to represent the images.

To achieve a satisfactory performance for CB-HRRS-IR, this paper focuses on extracting powerful features for better remote sensing image representation. Because a high resolution remote sensing image contains abundant information with a large image size, high dimensional feature vectors with hundreds or even thousands of codes are usually employed for the image representation [2,3]. However, regarding the similarity measurements of different images, especially in a large image database, high-dimensional features will increase the number of computation assumptions greatly. Thus, long feature codes with high dimensions for image representation will have a notable negative impact on the retrieval efficiency. In this paper, we propose and analyze Deep Compact Codes (DCCs) with low dimensions for remote sensing image representations to advance the image retrieval efficiency. Specifically, several schemes are experimentally implemented to learn the DCCs via Deep Convolutional Neural Networks (DCNNs) for remote sensing image representations. Additionally, the CNNs are trained from scratch using a large and high resolution remote sensing image archive that we collected. This archive will be made publicly available to other researchers. Our learned DCCs show a better representation for remote sensing image retrieval. This representation can highly improve the image retrieval performance with respect to both precision and efficiency. In addition, we explore a new method called the Feature Weighted Map (FWM) to assist in the visual understanding of deep features. The FWM can facilitate the process of determining the mechanisms underlying the efficacy of the proposed DCCs method. Furthermore, our proposed visualization method can also provide insights into the information that can be learned from the DCNN frameworks.

The remainder of this paper is organized as follows. In Section 2, we introduce the background and review the most relevant studies on remote sensing image retrieval. In Section 3, we introduce the image feature extraction methods, including the DCC learning schemes and PCA compression, and describe the evaluation methods from both quantitative and visual perspectives. In Section 4, we introduce the large remote sensing image archive, which will be released publicly to other researchers, and provide the experimental and analysis results. In Section 5, we describe the FWM. Finally, in Section 6, we draw conclusions regarding this work.

2. Background and Related Studies

Classical image features, such as spectral features [4,5], texture features [6–8], shape features [9,10] and morphological features [7] are the most common features used for remote sensing image representation. Great success with regard to high-resolution remote sensing image retrieval have been achieved using these global features. Compared with global features, local features are also good at representing remote sensing images with high spatial resolution, and they allow for the recognition of a greater range of objects and spatial patterns in small patches. Yang et al. [1] conducted the first investigation of the use of local invariant features for overhead image retrieval. Extensive experiments showed the effectiveness and practicability of local features for high resolution aerial imagery retrieval. Yang et al. [11] proposed a method that represents images using local features based on the typical Bag-of-Words (BoW) framework, which can improve the recognition performance of the remote sensing image retrieval process and reduce the burden of building the image index. Rather than extract either traditional global or local features, Wang et al. [12] and Bosilj et al. [13] explored new methods that can utilize both global and local features for remote sensing image retrieval. Additionally, recent studies have proposed methods that account for structure information in the image representations [14–16].

Although accurate features can be extracted via various methods for remote sensing image retrieval, they cannot be easily employed to describe a user's understanding of an image, which is significant to a user's intention. In other words, the semantic contents of an image cannot be well revealed by these features. To alleviate this issue, Liu et al. [17] proposed a region-level semantic mining approach for image presentation and constructed a uniform region-based depiction for each image by segmenting the images by region. Then, the semantic features were extracted using a

probabilistic method, which had good retrieval precision and recall. Wang et al. [18] proposed a remote sensing image retrieval scheme using image scene semantic matching. In addition, a prototype system that uses a coarse-to-fine retrieval scheme was implemented, and it had good retrieval accuracy. Recently, Linda et al. [19] presented a novel semantic mining and hashing method for remote sensing image retrieval, which showed good performance in their implemented retrieval system.

Most of the features mentioned in the above studies were low-level features that were individually designed, and they have been employed with a certain degree of success in remote sensing image retrieval. However, designing a stable and powerful feature representation for images could be a difficult task. In addition, remote sensing images with high resolution usually represent large geospatial scenes that contain abundant and complex visual contents. These factors can reduce the ability of low-level features to represent high-level abstract concepts in remote sensing images, which is known as the semantic gap between low-level features and high-level semantic content.

Recently, deep learning was shown to achieve considerable success in many tasks, including speech recognition [20,21], object recognition and detection [22–25] and natural language processing [26,27]. Inspired by such great success, high-level features extracted via deep learning have been introduced in the application of content-based high-resolution remote sensing image retrieval. Zhou et al. [28] utilized an unsupervised feature learning framework based on auto-encoder to map low-level feature descriptors to sparse feature representations for remote sensing image retrieval. Li et al. [2] also employed unsupervised multilayer feature learning and collaborative affinity metric fusion for remote sensing image retrieval. These methods can offer a higher-level feature representation of remote sensing images and outperform conventional features. However, the improvements are limited, and the unsupervised feature learning framework might not provide results that can be generalized because these frameworks are based on shallow networks that increase the difficulty of learning sufficiently powerful feature representations of remote sensing images. Moreover, the features learned by an unsupervised framework might require longer codes for image representation to achieve satisfactory retrieval results. This approach will obviously reduce the image retrieval efficiency. Napoletano [29] conducted an extensive evaluation of visual descriptors for the content-based retrieval of Remote Sensing (RS) images, including global, local, and CNN features. The results demonstrated that CNN-based and local features have the best performance in different retrieval schemes. Zhou et al. [30] investigated the extraction of features from both fully-connected and convolutional layers for remote sensing image retrieval. They [29,30] employed only CNN models and performed fine-tuning on a public remote sensing image dataset for feature extraction. Intensive comparisons were conducted to evaluate the performances of different models. Although these methods have achieved good performance in certain domains via DCNN frameworks, the learned deep features have not been sufficiently evaluated or described.

Visualization studies [23,31–33] have been conducted to better understand these deep features. Zeiler [23] developed deconvolutional networks to provide insights into the functions of intermediate feature layers and the operation of the classifier. However, deconvolutional networks do not always work well without max-pooling layers. Based on Zeiler's work, Springenberg [31] explored a guided backpropagation method that results in qualitative improvements. Zhou [34] developed a visualization method called class activation mapping that can localize objects. However, these methods only focus on the convolutional layers and ignore the fully connected layers, which play significant roles in feature representation. Dosovitskiy [32] and Mahendran [33] developed approaches to studying image representations by inverting deep features at different layers; however, these approaches show only image information rather than object-relating information preserved in the final deep features. Selvaraju [35] used the class-specific gradient information flowing into the final convolutional layer of a CNN to produce a coarse localization map of an object. However, this method can generate only a class-oriented visualization map based on the final classification score and is not applicable to non-classification tasks.

In this paper, we investigate and evaluate the performance of remote sensing image retrieval from a dimensional perspective. We analyze a series of different dimensional features that we call DCCs,

which are extracted by improved classical CNNs. In addition, we also perform a PCA to compress the high dimensional feature codes learned by the DCNNs, a strategy that is referred to as Deep Principal Component Analysis (DPCA) in the retrieval experiments. Furthermore, we explore a new method for visualizing deep features to provide a better understanding of our learned DCC features. Compared with the known archives [1,29,36], a high-resolution remote sensing image archive with a much larger scale is used to train the CNNs. The CNNs are trained from scratch to acquire a powerful representation of the remote sensing images and explore the performance of the DCCs in the task of CB-HRRS-IR.

The main contributions of this paper are as follows:

- We propose the extraction of DCCs for CB-HRRS-IR via two schemes. First, we extract a series of different dimensional DCCs that include a few tens to thousands of codes as the feature representation of remote sensing images. Second, PCA is introduced to compress the high-dimensional remote sensing image feature codes learned by traditional DCNNs. The lower-dimensional feature codes outperform the higher-dimensional ones, and the DCCs outperform the DPCA. In addition, we explore the FWM visualization method for deep features learned by DCNN frameworks, which can help us to better understand the differences between DCC features and the original deep features.
- Compared with the fine-tuning methods of former studies, we train all the DCNNs from scratch to explore the performance of the DCCs in CB-HRRS-IR based on a large-scale remote sensing image archive. In addition, this large-scale high-resolution remote sensing image archive will be made publicly available to other researchers. We expect that the archive can serve as a standardized public dataset in this field and can help to advance the research in the remote sensing field.

3. Methods

In this section, we first review the off-the-shelf DCNN frameworks and then introduce the DCC feature extraction schemes evaluated in our work. Next, we present the evaluation protocols for the experimental results. Finally, we introduce our proposed visualization approach, which is designed to provide a better understanding of deep features.

3.1. DCNN Framework

A traditional DCNN framework usually consists of several different types of layers, including convolutional layers, pooling layers, and fully connected layers (Figure 1). In a convolutional layer, a certain number of convolutional kernels are used to generate feature maps from the previous layer. A pooling layer is applied to reduce the spatial dimensions of the feature map via an average or max pooling operation. One or several fully connected layers follow a convolutional layer or a pooling layer and constitute the final part of the DCNN framework. Note that the pixel values of a feature map and a fully connected layer are usually mapped via an activation function, such as Rectified Linear Units (ReLUs) [22], Leaky ReLU (LReLU) [37] and the improved Parametric ReLU (PReLU) [38]. These activation functions can improve a CNN framework's nonlinearity and effectively expedite the convergence of the training procedure and avoid overfitting; thus, they highly boost the framework's generalization capacity.

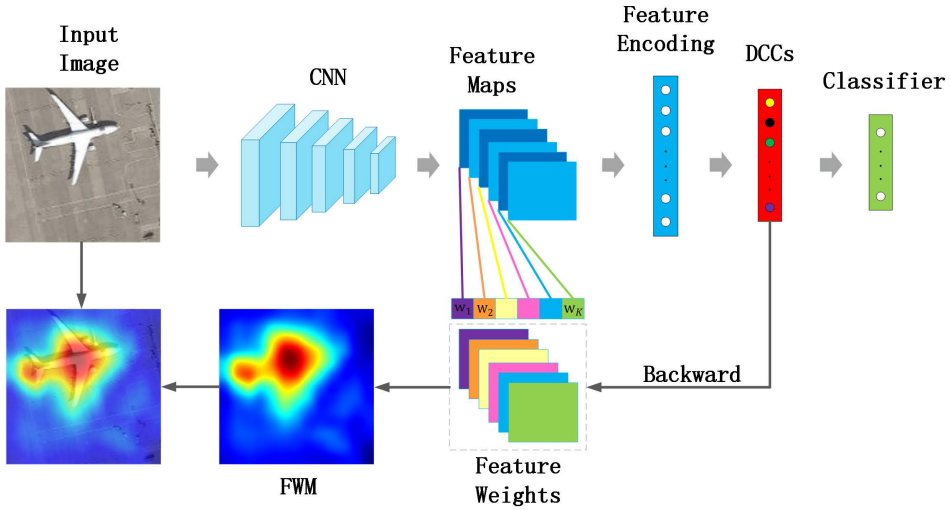

Figure 1. Framework of the DCC and FWM. An input image is mapped via the CNN to learn DCCs as image feature representations. The different colors of the DCC features indicate the different importance of each element in the feature space.

3.2. Feature Extraction

3.2.1. Feature Extraction Based on DCC

In general, a DCNN framework takes a raw image as input and processes it using a certain number of convolutional layers; then, it outputs feature maps of the original image in the final convolutional layer. The following fully connected layers then learn to convert the feature maps to a vector for image representation. In our opinion, the final fully connected layers can be regarded as an ordinary neural network for encoding the learned convolutional features. However, many studies have used high-dimensional codes (usually thousands of codes) in the fully connected layers for image representation for the tasks of object recognition, detection [22,23] and image retrieval [2,29,30]. Although these features are already rather compact compared with convolutional features, more compact features must be identified to further enhance the efficiency of the retrieval. Hinton [39] used neural networks for image data dimensionality reduction for the first time. Inspired by this work, we regard the fully connected layers in the DCNN framework to be ordinary neural networks that learn more compact codes (a few tens to thousands of codes) for image representation for the remote sensing image retrieval task.

In our experiments, two classical DCNN frameworks, i.e., Alexnet [22] and VGG-16 [40], are applied to compress the convolutional features. Alexnet includes five convolutional layers followed by three fully connected layers while VGG-16 contains 13 convolutional layers followed by three fully connected layers. The fully connected layers usually learn the high-level abstract feature representation of an image. Additionally, image features are often extracted from the second fully connected layer with high dimensionality. Based on our recognition process, the first fully connected layer (Fc1) encodes the learned feature map as a high dimensional feature vector. The second fully connected layer (Fc2) can then be used to learn more compact feature codes with lower dimensionality from Fc1. Finally, the compact feature codes are involved into a classifier in the third layer (Fc3). Therefore, the Fc2 layer is important for learning DCCs with different dimensionalities and can be regarded as a DCC learning layer (Figure 1). To evaluate the performance of DCC features in the application of remote sensing image retrieval, we set the dimensions of the DCC learning layer to 4096, 1024, 256, 64, and 32. Then, we extract a series of different dimensional deep compact feature codes to further explore the retrieval performance.

3.2.2. Feature Extraction Based on PCA

Although DCCs can be employed to replace the high-dimensional features extracted by the original DCNN frameworks, PCA, which is a classical data compression method with solid statistical foundations, is widely used in dimensionality reduction. To further evaluate the retrieval performance of the proposed DCC schemes, we also adopt PCA to compress the high-dimensional deep feature codes and compare the retrieval performance with that of the DCCs. Specifically, we have a set of n features $\{\boldsymbol{f}_1, \boldsymbol{f}_2, ..., \boldsymbol{f}_n\}, \boldsymbol{f}_i \in \mathbb{R}^D$, which are extracted by a raw DCNN framework and form the rows of the feature matrix $F \in \mathbb{R}^{D \times n}$. Our goal is to acquire a compressed feature matrix $F^{'} \in \mathbb{R}^{C \times n}$, where C denotes the length of the compressed feature codes. The basic principle of the PCA used to achieve this goal is to compress F via a projection operation, $F^{'} = U^T F$, where $U \in \mathbb{R}^{D \times C}$ is the projecting matrix. U can be obtained using the following objective function:

$$\begin{cases} max\ tr(U^T F F^T U) \\ s.t.\quad U^T U = I \end{cases} \tag{1}$$

The constraint $U^T U = I$ requires the projecting vectors to be orthogonal to one another in such a way that the compressed feature vectors are pairwise decorrelated. Similar to the DCC learning scheme, $\boldsymbol{f}_i$ is a deep feature that is extracted from the penultimate fully connected layer of an original DCNN framework. We compress $\boldsymbol{f}_i$ to yield shorter feature codes with the same dimensionality as of the DCCs. In the following sections, we use DPCA to refer to the feature codes that are compressed versions of the original deep features of the PCA method.

3.3. Evaluation Methods

3.3.1. Quantitative Evaluation

For the similarity measurements, three state-of-the art distances, which are the most commonly used for image retrieval, are applied: Manhattan distance, Euclidean distance and cosine distance. For the sake of efficiency, we adopt the Manhattan distance to identify the images that are similar to the query. Regarding the retrieval performance, the Precision (P), Recall (R) and mean Average Precision (mAP) are often employed to assess the retrieval results. Precision is defined as the fraction of the retrieved relevant images with respect to the query image, and recall is defined as the ratio of the number of retrieved relevant images to the total number of images that are relevant to the query image. Usually, only the *top-k* retrieved results are evaluated to determine their precision. The fraction of true relevant images in the *top-k* results (*P@k*) is calculated as follows:

$$P(k) = \frac{\sum_{i=1}^{k} \sigma(i)}{k} \tag{2}$$

where $\sigma(i)$ indicates the relevance between a query q and the i-th ranked retrieved image. Here, $\sigma(i) \in \{0, 1\}$ is 1 if the i-th item is a relevant image and 0 otherwise. To assess the performance of the ranked retrieval results, an interpolated recall-precision curve can be plotted to compare the differences and determine the comprehensive performance of the retrieval schemes. Given a set of Q queries, the mAP can be defined by calculating the Average Precision (AP) for all queries:

$$mAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{3}$$

where the AP for each query q is defined as follows:

$$AP = \frac{\sum_{k=1}^{R} P(k)\sigma(k)}{\sum_{j=1}^{R} \sigma(j)} \tag{4}$$

where R represents the size of the test dataset.

3.3.2. Visual Evaluation

The point of our explored visualization method is to extract image features from the penultimate fully connected layer and then to employ the backpropagation algorithm to map back the feature codes onto the convolutional feature layer, which yields the FWM. Specifically, for an image, the output of the fully connected layer is seen as a feature vector, and each code of the vector can be regarded as the importance of the corresponding dimensionality in the feature space. Therefore, we backpropagate the extracted feature as weights of the convolutional layers, and a weighted sum of the convolutional feature maps is used to generate the final FWM. Let $A_k(x, y)$ represent the activation of the k-th feature map of the last convolutional layer at position (x, y). To obtain the importance of a feature code at the d-th dimension f^d, which is learned from the activation $A_k(x, y)$ of a feature map, we first calculate the gradient of f^d with respect to $A_k(x, y)$:

$$g_k^d(x, y) = \frac{\partial f^d}{\partial A_k(x, y)} \tag{5}$$

Therefore, the whole contribution of $A_k(x, y)$ to the final feature can be calculated as follows:

$$G_k(x, y) = \sum_{d}^{D} g_k^d(x, y) \tag{6}$$

where D is the dimensionality of the output feature. Thus, for every activation at position (x, y) of a feature map, we can obtain its weight with respect to the final extracted feature using $G_k(x, y)$. However, the information contained in the final feature is not pure because of the image quality; thus, noise information is contained in the feature codes. As a result, this noise is also projected back to the weight of an activation $A_k(x, y)$ via the above operations. Considering this situation, we adopt the average weight of $A_k(x, y)$ as the final weight of the k-th feature map:

$$w_k = \frac{1}{n} \sum_{(x,y)} G_k(x, y) \tag{7}$$

where n is the number of pixels of the feature map. Usually, several feature maps correspond to the channels of the last convolutional layer. To generate the FWM for visualization, the weighted sum of these feature maps is calculated as follows:

$$FWM = w_k A_k(x, y) \tag{8}$$

Based on this method, we can visualize the information that is contained in the feature maps and preserved in the final feature codes. This approach helps us further understand the learned DCC feature codes and compare them to the traditional codes. Thus, this visualization method is feature oriented and can be generalized to any layer of a CNN framework to provide a representation of the nature of the CNN framework.

4. Experimental Section

4.1. Dataset

Extensive evaluations of retrieval performance were conducted using a large-scale high-resolution remote sensing image dataset composed of 25 classes of different scenes/objects. For each class, 500 images with the size of 256 by 256 pixels were manually collected, primarily from the *World Map* (World Map web site: http://map.tianditu.com/map/index.html.) website. The image classes are as follows: agricultural, airport, basketball court, bridge, building, container, fishpond, footbridge, forest, greenhouse, intersection, oiltank, overpass, parking lot, plane, playground, residential, river, ship, solar power area, square, tennis court, water, wharf, and workshop. Specifically, the image resolution of each class is 0.6 m per pixel, except for the airport class. Most of the images were collected from the 18-level remote sensing image found on the *World Map*. However, an image that is 256 by 256 pixels might be too small to contain a large remote sensing scene, such as an airport or an overpass. As a compromise, we collected airport images with a 2.4 m per pixel resolution that include 16-level remote sensing images from *World Map*; these images can clearly reflect the properties of an airport. For the overpass, we collected corresponding images with a resolution of 1.2 m that contain the main parts of an overpass. Additionally, a small number of plane images were collected from *Google Maps* as a supplement; the resolution was the same as that for the images from *World Map*. All the collected images are in the Red-Green-Blue (RGB) color space. Four samples of each class are shown in Figure 2. As explained above, our dataset contains large-scale remote sensing images that vary in resolution, image size, and source. Each of these factors pose challenges to the comprehensive retrieval performance of our experiments. This remote sensing image dataset will be made publicly available to other researchers, and we expect that it will greatly promote research in the remote sensing community, including research related to remote sensing image classification, scene classification, object detection/recognition, and image retrieval.

4.2. Experimental Setup

As introduced in the previous section, we evaluated the retrieval effect using the popular Alexnet [22] and VGG-16 [40] CNNs, and we set the penultimate fully connected layer to be of various dimensionalities (4096, 1024, 256, 64 and 32) to obtain the DCCs. For the dataset, we randomly selected 250 images for each class from the whole dataset as a training dataset and then used the remainder as the test dataset. For the training dataset, we randomly selected 200 images of each class as training samples with the remaining 50 images of each class being used for the validation dataset. Note that we only selected less than half of the images to train the CNNs, which is different from [30], who adopted most of their images from the dataset for the CNNs training. Thus, a situation in which a small number of the samples is used for training can result in overfitting because of the large-scale parameters in the CNNs. To overcome this problem, we employed simple data augmentations to enrich the training dataset. Specifically, the form of the data augmentation consisted of the generation of image horizontal reflections and rotations with degrees of 90, 180 and 270. Compared with previous studies [2,29,30] that trained the CNN models by fine-tuning the pretrained CNN models based on natural images, we trained the CNNs from scratch using high-resolution remote sensing images to avoid the latent differences between the natural images and remote sensing images.

For the required input dimensionality, all images were resized to 227 by 227 pixels for Alexnet and 224 by 224 pixels for VGG-16, as well as their corresponding transformation networks. In addition, the mean values were extracted from the training samples. Following the previous works [22,23], the weights were set using a Gaussian distribution for Alexnet and "Xavier" for VGG-16. We set the learning rate to 0.01 for Alexnet and 0.001 for VGG-16. The batch size was set to 256 for Alexnet and 48 for VGG-16. For a fair comparison, the transformation CNNs were initialized in a manner similar to

that of their corresponding original CNNs. All experiments were conducted using the Ubuntu-16.04 system and two Nvidia GTX Titan X GPUs with 12GB RAM.

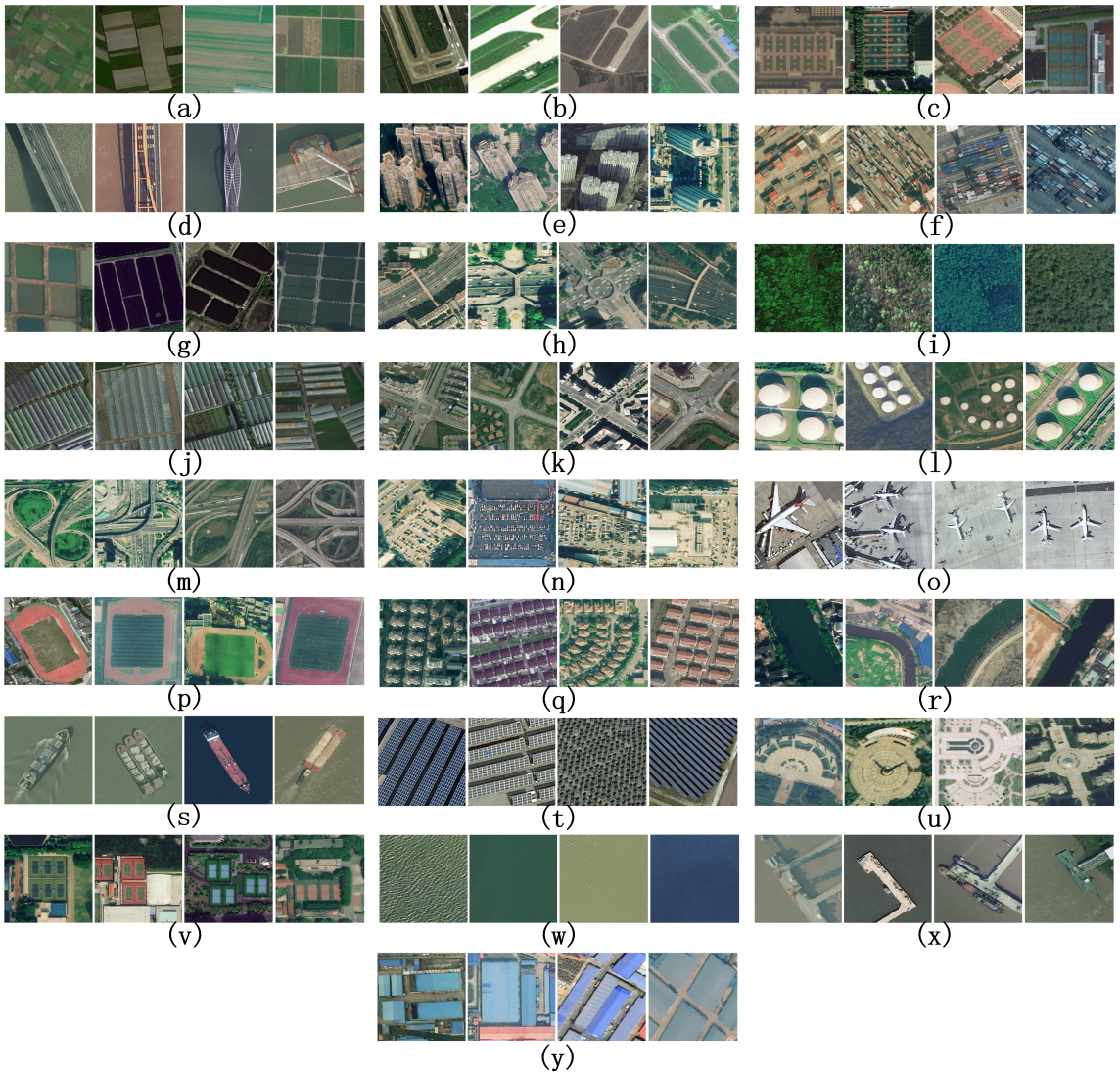

Figure 2. Ground truth dataset containing 500 images of each class. Four samples of the following classes are shown: (**a**) agricultural; (**b**) airport; (**c**) basketball court; (**d**) bridge; (**e**) building, (**f**) container; (**g**) fishpond; (**h**) footbridge; (**i**) forest; (**j**) greenhouse; (**k**) intersection; (**l**) oiltank; (**m**) overpass; (**n**) parking lot; (**o**) plane; (**p**) playground; (**q**) residential; (**r**) river; (**s**) ship; (**t**) solar power area; (**u**) square; (**v**) tennis court; (**w**) water; (**x**) wharf, and (**y**) workshop.

4.3. Results and Analyses

In this section, we compare the performances of the DCC and DPCA methods for different dimensions. The top-100 retrieval precision results of each class for different methods are evaluated. The top-150 retrieval precision results for each class are also shown for further evaluation. For convenience, we use "ADCC# "to represent the names of the DCC frameworks: it denotes DCCs with a dimensionality of # learned using the "Alexnet" framework. The same method is also applied to the corresponding VGG frameworks. Note that the length of the features extracted from the original Alexnet and VGG frameworks is 4096. Similarly, we use "APCA#" to represent the names of the DPCA learning methods: it denotes image feature codes learned by the original "Alexnet"

framework and then compressed to dimension of # via the PCA method. A similar method is also used for the corresponding VGG frameworks.

4.3.1. Retrieval Performance of DCCs

In this section, the image retrieval performance of the DCC method is evaluated. The top-100 retrieval precision results for each class are shown in Table 1. In Table 1, there are two results reported in bold in each row. The first result is the best retrieval result for Alexnet and the corresponding DCC models; the second one is the best retrieval result for VGG and the corresponding models. As shown in Table 1, most of the best results for Alexnet and its DCC models are obtained using ADCC64 and ADCC32, and all the best results are obtained using our DCC models. Specifically, the findings indicate that precision is greatly increased by our DCC models for all classes compared with results of the original Alexnet model. The river class in the ADCC64 model shows the largest increase in precision, with a significant improvement of nearly 20%. Moreover, nearly all our DCC models achieve obvious improvements in the retrieval results for each class compared with the results of the original Alexnet model. For VGG and its corresponding DCC models, most of the best results are achieved using VDCC64, and there is also a high improvement in the retrieval precision for each class compared with the original VGG model. Several of the best retrieval results are not obtained using VGG64, and the precision achieved using VDCC64 is comparable with the best results. Additionally, the best improvement is observed for the wharf class, for which the precision increases from 51.02% to 75.32%. Thus, the precision of the VDCC64 models is more than 24% higher than that of the original VGG model. Compared with the ADCC32 model, which achieved several of the best retrieval results, the VDCC32 model does not generate the best retrieval results for any class, and certain classes appear to show decreases in retrieval precision. Nevertheless, the VDCC32 model achieves prominent improvements in the retrieval precision for 60% of the classes and certain improvements in retrieval precision are noticeable compared with the results of the original VGG model, such as for the wharf and building classes, which showed precision improvements of 18.66% and 14.95%, respectively. Table 2 shows the top-150 retrieval precision results for all classes. The retrieval performance based on the P@150 values shows a similar trend as that based on the P@100 values. The greatest improvements in precision are observed for the river class using the ADCC64 model and the wharf class using the VDCC64 model. Taken together, all our DCC models can generate significant improvements in the retrieval precision compared with their corresponding original frameworks.

For Alexnet and its DCC models, several of the best results are obtained using ADCC32, and the corresponding results obtained using ADCC64 are similar to those of ADCC32. For the VDCC32 model, a sharp decline in the precision is observed compared with that of the VDCC64 model. These results indicate that image features with a dimensionality of 64 may be the most optimal for adapting to the image representation. To comprehensively assess the performances of different models, we compare the model accuracies of these methods, which are calculated according to the mean of retrieval precisions of each class achieved by the CNN model. The model accuracies based on the P@100 and P@150 values are listed in Tables 3 and 4, respectively. Additionally, the best results are reported in bold. Tables 3 and 4 show that the model accuracy clearly increases as the feature dimensionality is reduced from 4096 to 64. ADCC64 and VDCC64 achieve the best results at both the P@100 and P@150 levels, and remarkable accuracy improvements can be observed. Specifically, at the P@100 level, ADCC64 and VDCC644 show accuracy improvements of 8.81% and 8.15% compared with the results of the corresponding original CNN frameworks, respectively. At the P@150 level, ADCC64 and VDCC644 show accuracy improvements of 9.37% and 9.06% compared with the results of the baseline CNN frameworks, respectively. These findings are in accordance with the class precision results shown in Tables 1 and 2. In Tables 3 and 4, each row shows approximate accuracy improvements as the dimensions of the DCCs decrease from 4096 to 1024, 256 and 64. Thus, the efficacy of our proposed DCC method is demonstrated by the performances of different types of CNN frameworks. Regarding the 32-dimensional feature codes, both ADCC32 and VDCC32 show decreases in model accuracy compared

with the corresponding 64-dimensional feature codes, especially VDCC32. This finding indicates that 64-dimensional feature codes are more effective for image representation than feature codes with other dimensions, which confirms our former hypothesis. Moreover, even when decreases in accuracy are observed for the 32-dimensional features compared with 64-dimensional features, the ADCC32 and VDCC32 models still achieve significantly higher accuracies than those of the Alexnet and VGG models, respectively.

Specifically, a comparison of the same dimensional feature codes from different frameworks shows that VGG and its corresponding DCC models achieve higher accuracies than those of Alexnet and its corresponding DCC models. This finding makes sense because VGG and its corresponding DCC models are deep CNN frameworks, while Alexnet and its corresponding DCC models are shallow CNN frameworks. Nevertheless, a comparison of the results of the DCC models of Alexnet with those of the original VGG model shows that the accuracies of the DCC models are considerably higher than those of the VGG model, especially for the Alexnet64 and Alexnet32 models, which present accuracy improvements of 6.89% and 6.30%, respectively. These findings demonstrate that the feature codes with low dimensionality learned by our DCC models have more powerful image representation abilities compared with the high-dimensional features codes. This finding reveals that high-dimensional features learned by traditional CNN frameworks do not always have the best image representation abilities, whereas the lower-dimensional features that can be learned by our DCC method can greatly improve upon the performance of the original frameworks. More importantly, these results indicate that our proposed DCC method can achieve a better performance using a shallow CNN framework rather than a deep CNN framework. The superiority of this approach is obvious. Our proposed DCC method is easy to use, and it can also reduce the training time requirements and improve the convenience of the applications. This discovery can also be employed for many other tasks, such as image classification and object detection, to further advance the performance and efficiency of the method because of the shorter and more powerful feature representation capacity.

Table 1. P@100 values obtained using the models based on the Alexnet, VGG and corresponding DCC frameworks.

Class	Alexnet	ADCC1024	ADCC256	ADCC64	ADCC32	VGG	VDCC1024	VDCC256	VDCC64	VDCC32
agricultural	0.5241	0.5448	0.6005	0.6770	**0.6882**	0.6908	0.6575	0.7082	**0.7154**	0.5986
airport	0.5979	0.6504	0.7274	0.7326	**0.7414**	0.5594	0.6037	0.6771	**0.7116**	0.6675
basketball court	0.4384	0.5227	0.5111	**0.5611**	0.5322	0.3478	0.3819	0.4186	**0.4509**	0.4352
bridge	0.8344	0.8756	0.9004	**0.9015**	0.8900	0.7831	0.8612	0.8776	**0.8953**	0.8441
building	0.5965	0.6491	0.6502	**0.6728**	0.6687	0.5119	0.5944	0.6236	**0.6829**	0.6614
container	0.6688	0.7232	0.7752	**0.7896**	0.7859	0.7718	0.7863	0.8016	**0.8057**	0.7821
fishpond	0.8289	0.8703	**0.8810**	0.8748	0.8590	0.8160	0.8322	0.8300	**0.8541**	0.8450
footbridge	0.6231	0.6723	0.6967	0.7304	**0.7350**	0.7177	0.7404	0.7766	**0.8113**	0.6802
forest	0.9126	0.9057	0.9057	0.9272	**0.9290**	0.9407	0.9419	0.9495	**0.9502**	0.9075
greenhouse	0.9234	0.9316	**0.9446**	0.9415	0.9444	0.9458	0.9497	**0.9559**	0.9545	0.9132
intersection	0.6707	0.7174	0.7850	**0.8147**	0.7996	0.8196	0.8379	0.8375	**0.8462**	0.8154
oiltank	0.8043	0.8498	0.8879	0.8810	**0.8951**	0.8223	0.8571	0.9064	**0.9128**	0.8469
overpass	0.6624	0.6903	0.7793	0.7825	**0.7870**	0.7768	0.8016	0.8288	**0.8696**	0.8230
parking lot	0.5376	0.5815	0.6198	**0.6630**	0.6549	0.5711	0.6256	0.6720	**0.7149**	0.6664
plane	0.8502	0.8666	0.8819	0.8729	**0.8847**	0.7719	0.8298	0.8513	**0.8980**	0.8534
playground	0.7081	0.7162	0.7556	**0.7709**	0.7475	0.7097	**0.7464**	0.7406	0.7426	0.6855
residential	0.7020	0.7459	0.7194	**0.7466**	0.7422	0.7307	0.7439	0.7756	**0.7792**	0.7663
river	0.4669	0.5080	0.6092	**0.6655**	0.6416	0.5115	0.5874	0.5966	**0.6575**	0.5987
ship	0.8885	0.8799	**0.8982**	0.8936	0.8654	0.9245	0.9208	**0.9380**	0.9055	0.8886
solar power area	0.7502	0.7732	0.7757	**0.8214**	0.8164	0.8157	0.8235	**0.8458**	0.8374	0.7908
square	0.5082	0.5187	0.6074	0.6435	**0.6570**	0.5153	0.5323	0.6053	**0.6634**	0.6362
tenniscourt	0.4515	0.5043	0.5366	0.5928	**0.6103**	0.4915	0.5460	0.6249	**0.6257**	0.3984
water	0.8726	0.9008	0.9003	**0.8920**	0.8609	**0.9249**	0.9053	0.9149	0.9142	0.8043
wharf	0.6476	0.7325	0.7602	**0.8004**	0.7858	0.5102	0.6530	0.7272	**0.7532**	0.6968
workshop	0.8676	0.8703	0.8802	**0.8899**	0.8705	0.8378	0.8658	0.8730	**0.9033**	0.8709

Table 2. P@150 values obtained using the models based on the Alexnet, VGG and corresponding DCC frameworks.

Class	Alexnet	ADCC1024	ADCC256	ADCC64	ADCC32	VGG	VDCC1024	VDCC256	VDCC64	VDCC32
agricultural	0.4722	0.4981	0.5591	0.6337	**0.6461**	0.6420	0.6106	0.6628	**0.6696**	0.5462
airport	0.5531	0.6121	0.6865	0.6931	**0.6980**	0.5018	0.5523	0.6232	**0.6591**	0.6241
basketball court	0.4172	0.4955	0.4827	**0.5293**	0.5048	0.3246	0.3603	0.4003	**0.4282**	0.4136
bridge	0.8001	0.8628	0.8856	**0.8915**	0.8732	0.7309	0.8264	0.8580	**0.8727**	0.8140
building	0.5496	0.6130	0.6198	**0.6334**	0.6259	0.4697	0.5533	0.5849	**0.6455**	0.6284
container	0.6094	0.6767	0.7472	**0.7547**	0.7502	0.7127	0.7436	0.7671	**0.7709**	0.7311
fishpond	0.7986	0.8474	0.8636	**0.8459**	0.8309	0.7817	0.8063	0.7960	**0.8250**	0.8110
footbridge	0.5793	0.6289	0.6565	0.6879	**0.6971**	0.6777	0.6974	0.7332	**0.7713**	0.6369
forest	0.8868	0.8894	0.8923	0.9165	**0.9216**	0.9281	0.9340	**0.9370**	0.9362	0.8834
greenhouse	0.9032	0.9144	**0.9278**	0.9138	0.9241	0.9329	0.9360	**0.9440**	0.9423	0.8972
intersection	0.6257	0.6824	0.7522	**0.7729**	0.7585	0.7790	0.7979	0.8031	**0.8048**	0.7796
oiltank	0.7667	0.8228	**0.8650**	0.8571	0.8637	0.7755	0.8186	0.8757	**0.8782**	0.8170
overpass	0.6150	0.6525	0.7474	**0.7497**	0.7464	0.7295	0.7642	0.7968	**0.8364**	0.7926
parking lot	0.4980	0.5463	0.5839	**0.6250**	0.6129	0.5241	0.5841	0.6203	**0.6715**	0.6221
plane	0.8074	0.8335	**0.8512**	0.8437	0.8453	0.7107	0.7866	0.8119	**0.8635**	0.8119
playground	0.6707	0.6802	0.7240	**0.7309**	0.7003	0.6617	0.7017	**0.7027**	0.6997	0.6330
residential	0.6578	0.7149	0.6900	**0.7125**	0.6944	0.6838	0.7003	**0.7432**	0.7413	0.7366
river	0.4204	0.4607	0.5687	**0.6142**	0.5853	0.4581	0.5360	0.5506	**0.6089**	0.5363
ship	0.8631	0.8638	**0.8875**	0.8682	0.8401	0.8934	0.8960	**0.9240**	0.8873	0.8650
solararea	0.7139	0.7447	0.7513	**0.7827**	0.7786	0.7685	0.7862	**0.8131**	0.8040	0.7535
square	0.4662	0.4827	0.5676	0.6012	**0.6045**	0.4580	0.4841	0.5566	**0.6179**	0.5989
tennis court	0.4130	0.4757	0.5078	0.5553	**0.5726**	0.4426	0.4941	0.5806	**0.5874**	0.3786
water	0.8554	0.8883	**0.8909**	0.8830	0.8487	**0.9235**	0.9036	0.9145	0.9106	0.7599
wharf	0.6024	0.6945	0.7264	**0.7689**	0.7502	0.4549	0.6035	0.6823	**0.7209**	0.6458
workshop	0.8387	0.8393	0.8578	**0.8614**	0.8401	0.7969	0.8308	0.8378	**0.8746**	0.8301

Table 3. Retrieval accuracies of the models based on Alexnet, VGG and corresponding DCC frameworks according to the P@100 values.

Model	Accuracy	Model	Accuracy
Alexnet	69.35%	VGG	71.27%
ADCC1024	72.80%	VDCC1024	74.50%
ADCC256	75.96%	VDCC256	77.43%
ADCC64	**78.16%**	VDCC64	**79.42%**
ADCC32	77.57%	VDCC32	73.91%

Table 4. Retrieval accuracies of the models based on Alexnet, VGG and corresponding DCC frameworks according to the P@150 values.

Model	Accuracy	Model	Accuracy
Alexnet	65.54%	VGG	67.05%
ADCC1024	69.68%	VDCC1024	70.83%
ADCC256	73.17%	VDCC256	74.08%
ADCC64	**74.91%**	VDCC64	**76.11%**
ADCC32	74.05%	VDCC32	70.19%

For further evaluation, we employ a confusion matrix to show the classification results for 64- and 4096-dimensional features extracted from different models, as shown in Tables 5–8. For simplicity, let C1, C2, ..., C25 represent agricultural, airport, basketball court, bridge, building, container, fishpond, footbridge, forest, greenhouse, intersection, oiltank, overpass, parking lot, plane, playground,

residential, river, ship, solar power area, square, tennis court, water, wharf and workshop, respectively. Each column in Tables 5–8 corresponds to the prediction result, while each row represents the actual class. Note that the last column and row correspond to the classification and prediction precision, respectively. In addition, the overall accuracy is reported in bold in the right-bottom cell. As seen from Tables 5 and 6, even though there are slight decreases in classification precision for several classes, most of the classes experienced improvements in classification precision when the ADCC64 model was used. In addition, the precision improvements are more obvious compared to the decreases in classification precision, such as for river (C18), square (C21), tennis court (C22), which achieved a 9.60%, 6.80%, and 7.60% higher classification precision, respectively, when our DCC method was used, respectively. In addition, the prediction precision of each class shows a similar trend. The intersection class achieved a 10.70% improvement in prediction precision when the ADCC64 model was used. For the VGG and VDCC64 models (Tables 7 and 8), most classes experienced precision improvements in both classification and prediction when our DCC method was used. However, the playground (C16) class suffered a 5.60% decrease in classification precision when the VDCC64 model was used. This result occurred because some playground samples are more easily classified as basketball court (C3) samples, as shown in Table 8. In addtion, some tennis court (C22) samples are also classified wrongly as basketball court samples. This led to a 3.78% decrease in the prediction precision of the basketball court class. The main reason for this phenomenon is that there is a certain similarity between the backgrounds of the tennis court, playground and basketball court samples to some extent. The VDCC64 model is more capable of discriminating the basketball court class, as the classification precision for the basketball court class using this model experienced a 10.80% improvement. This result implies that the VDCC64 model can learn to discriminate the basketball court class from other classes. It also reveals that VDCC64 has some deficiencies in discriminating similar classes. Nevertheless, our DCC method can achieve a higher overall classification accuracy. For the ADCC64 model, the overall classification accuracy is 86.56%, which is nearly 2.00% higher than that of the original Alexnet model (84.58%). In addition, the VGG64 model achieves an approximately 1.50% improvement in overall classification accuracy (88.37%) compared with the original VGG model (86.88%). In general, our DCC models can achieve better classification results for most classes.

Table 5. Confusion matrix of classification results based on Alexnet.

Class	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	P(%)
C1	194	5	1	0	0	0	5	2	2	4	1	0	9	0	1	2	0	5	0	5	3	4	6	1	0	77.60
C2	3	198	1	1	4	0	1	1	0	3	5	3	10	1	7	1	2	1	0	3	5	0	0	0	0	79.20
C3	3	0	147	0	4	1	0	3	1	0	0	2	0	6	0	32	5	1	0	0	5	38	0	2	0	58.80
C4	2	0	0	231	0	0	3	0	0	0	0	1	1	0	2	0	0	0	3	0	0	0	2	4	1	92.40
C5	0	1	1	0	193	3	1	6	0	0	3	2	2	15	0	1	12	2	0	0	3	4	0	1	0	77.20
C6	1	1	1	0	0	211	0	0	0	0	1	1	0	19	3	0	6	0	0	3	2	0	0	0	1	84.40
C7	0	0	0	0	0	0	231	1	0	1	1	0	1	0	2	0	0	6	2	1	0	3	0	1	0	92.40
C8	1	1	0	0	6	1	0	190	0	0	14	1	11	18	0	0	1	3	0	0	2	0	0	0	1	76.00
C9	0	1	0	0	0	0	0	1	241	0	0	0	0	1	0	0	0	1	0	3	0	0	2	0	0	96.40
C10	0	0	0	0	0	0	0	0	0	248	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	99.20
C11	1	0	0	0	0	0	0	2	2	0	229	0	5	3	1	0	1	0	0	3	2	0	1	0	0	91.60
C12	0	1	0	0	1	0	0	1	0	0	1	231	2	4	1	0	0	0	0	1	4	0	0	0	3	92.40
C13	1	1	0	0	0	0	1	3	0	0	16	0	221	0	0	1	0	1	0	5	0	0	0	0	0	88.40
C14	1	0	2	0	12	2	0	3	2	0	4	2	0	211	0	0	2	0	0	0	3	1	0	1	4	84.40
C15	0	3	0	0	0	1	0	0	0	0	0	0	0	0	240	0	0	0	0	0	2	0	0	4	0	96.00
C16	0	1	22	0	0	0	2	2	0	0	0	1	0	2	2	212	0	0	0	0	4	2	0	0	0	84.80
C17	0	0	0	0	9	2	0	1	4	1	6	0	0	13	0	0	211	1	0	0	0	2	0	0	0	84.40
C18	11	4	2	3	8	2	3	7	6	0	14	0	6	5	2	2	1	170	0	0	0	3	0	1	0	68.00
C19	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	242	0	0	0	1	5	0	96.80
C20	3	5	0	1	0	2	2	2	0	1	0	1	2	3	0	1	2	0	0	224	0	0	1	0	0	89.60
C21	10	3	0	0	10	7	1	2	1	1	5	1	2	18	5	6	1	1	0	0	172	4	0	0	0	68.80
C22	2	0	20	0	4	2	1	6	0	0	3	2	5	9	0	7	6	3	0	0	5	174	0	0	1	69.60
C23	7	0	0	0	0	0	0	0	11	0	0	0	0	0	1	0	0	0	0	6	0	0	223	0	2	89.20
C24	0	0	0	4	1	3	1	0	0	1	1	0	0	0	5	0	0	4	22	0	1	1	0	206	0	82.40
C25	0	1	3	1	0	2	1	0	0	0	0	0	0	1	0	0	0	0	0	3	1	0	0	1	236	94.40
P(%)	80.83	87.61	73.50	95.85	76.59	88.28	91.30	81.55	88.93	95.38	75.33	93.15	79.50	64.13	87.91	79.70	84.40	85.43	89.96	87.16	80.37	73.73	94.49	90.75	94.78	**84.58**

Table 6. Confusion matrix of classification results based on ADCC64.

Class	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	P(%)
C1	204	4	2	0	0	0	5	5	0	3	0	0	6	1	1	1	0	3	0	5	2	3	5	0	0	81.60
C2	4	204	2	0	1	1	0	0	0	2	3	2	13	1	5	0	0	3	0	1	6	2	0	0	0	81.60
C3	0	0	161	0	5	0	0	1	0	0	0	0	0	7	0	19	4	0	0	0	7	43	0	3	0	64.40
C4	2	2	0	233	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	8	0	93.20
C5	0	0	4	1	189	3	0	7	0	0	1	0	2	17	0	0	14	2	0	0	7	3	0	0	0	75.60
C6	0	1	0	1	0	214	0	0	0	0	0	0	0	21	1	0	4	0	0	4	1	0	0	2	1	85.60
C7	1	1	0	0	0	0	235	1	1	2	0	0	1	0	0	0	0	6	2	0	0	0	0	0	0	94.00
C8	2	1	1	0	9	0	0	204	0	0	8	1	5	14	0	0	1	2	0	1	0	1	0	0	0	81.60
C9	1	0	0	0	0	0	0	0	239	0	0	0	0	1	0	0	1	1	0	1	0	0	6	0	0	95.60
C10	0	0	0	0	0	0	0	0	0	247	0	0	1	0	0	0	0	0	0	2	0	0	0	0	0	98.80
C11	1	0	1	0	1	1	0	2	0	0	228	1	9	3	1	0	0	0	0	1	1	0	0	0	0	91.20
C12	0	0	0	0	0	1	0	0	0	0	0	233	0	1	2	0	0	1	0	2	6	1	0	0	3	93.20
C13	3	2	0	0	0	0	2	5	0	0	10	0	224	0	0	1	0	1	0	2	0	0	0	0	0	89.60
C14	0	0	1	0	7	3	0	3	0	0	1	2	2	215	0	0	0	2	0	2	8	2	0	1	1	86.00
C15	0	6	0	0	0	0	0	0	0	0	0	0	0	0	236	0	0	1	0	0	4	0	0	3	0	94.40
C16	0	1	31	0	1	0	1	0	0	0	0	1	0	0	0	210	0	0	0	0	5	0	0	0	0	84.00
C17	0	0	0	0	13	3	0	2	0	0	5	0	2	11	0	0	210	1	0	0	0	2	0	1	0	84.00
C18	10	2	3	0	6	2	2	10	3	0	3	0	6	3	0	0	1	194	0	0	0	5	0	0	0	77.60
C19	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	243	0	0	0	1	4	0	97.20
C20	4	2	0	0	0	2	1	0	2	1	0	0	0	0	0	0	2	0	0	232	3	0	0	1	0	92.80
C21	7	2	2	0	4	0	1	3	0	0	3	1	2	21	6	4	0	0	0	0	189	4	0	1	0	75.60
C22	2	0	19	0	4	1	0	4	0	0	3	0	3	6	0	3	3	3	0	0	6	193	0	0	0	77.20
C23	9	0	0	0	0	0	0	0	6	1	0	0	0	0	0	0	0	0	0	7	0	0	225	0	2	90.00
C24	1	0	0	2	3	3	1	0	0	0	0	0	0	2	3	0	0	3	19	0	0	0	0	213	0	85.20
C25	0	0	0	2	0	2	1	0	0	0	0	0	0	1	1	0	0	1	0	4	1	2	0	0	235	94.00
P(%)	81.27	89.47	70.93	97.49	77.78	90.68	94.38	82.59	94.84	96.48	86.04	96.68	81.16	66.15	91.83	88.24	87.50	86.61	92.05	86.25	76.83	73.95	94.94	89.87	97.11	**86.56**

Table 7. Confusion matrix of classification results based on VGG.

Class	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	P(%)
C1	206	6	1	0	0	0	8	3	2	3	1	0	0	0	0	1	3	6	0	1	2	3	3	0	1	82.40
C2	8	202	3	2	2	0	0	2	0	1	1	4	7	1	3	1	2	4	0	0	7	0	0	0	0	80.80
C3	0	3	131	0	9	2	2	0	0	0	2	0	1	4	1	35	6	5	0	0	9	37	0	2	1	52.40
C4	3	2	0	234	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	6	1	93.60
C5	0	0	6	0	202	1	0	1	0	0	0	4	2	5	0	0	14	6	0	0	5	3	0	1	0	80.80
C6	0	0	0	0	3	221	0	3	0	0	0	2	0	6	2	0	6	0	0	0	4	0	0	2	1	88.40
C7	4	1	1	0	1	0	231	0	0	1	0	1	2	0	0	1	0	5	0	0	0	0	0	2	0	92.40
C8	1	1	0	0	7	1	1	219	0	0	1	0	2	5	0	1	1	4	0	0	4	0	0	1	1	87.60
C9	1	0	0	0	0	0	0	0	245	0	1	0	0	0	0	0	1	1	0	0	0	0	1	0	0	98.00
C10	2	0	0	0	0	0	1	0	0	246	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	98.40
C11	2	1	1	0	0	1	0	0	0	0	233	0	7	1	1	0	1	1	0	0	0	1	0	0	0	93.20
C12	0	0	0	0	0	1	0	0	0	0	0	244	0	1	1	0	0	0	0	0	1	1	0	1	0	97.60
C13	0	0	0	0	0	0	0	1	0	0	7	0	237	1	0	1	1	1	0	0	0	1	0	0	0	94.80
C14	0	0	5	0	12	5	0	3	0	0	2	1	0	203	2	0	2	2	0	0	5	4	0	1	3	81.20
C15	0	4	0	0	0	3	0	0	0	0	0	1	0	0	235	0	0	1	1	0	4	0	1	0	0	94.00
C16	0	2	19	0	0	0	2	0	0	0	0	1	0	0	0	220	0	1	0	0	4	1	0	0	0	88.00
C17	0	0	0	0	8	4	0	1	1	0	1	1	0	4	0	0	224	3	0	0	1	2	0	0	0	89.60
C18	18	7	1	0	2	2	3	4	3	0	4	0	2	3	0	1	4	187	0	0	1	4	0	2	2	74.80
C19	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	243	0	0	0	0	5	0	97.20
C20	5	1	0	1	0	1	0	3	0	5	1	0	2	0	3	0	2	0	1	222	3	0	0	0	0	88.80
C21	10	9	4	0	6	5	0	3	0	0	2	4	0	13	4	2	1	1	0	0	181	5	0	0	0	72.40
C22	2	0	26	0	6	1	0	3	1	2	1	2	0	1	1	2	6	5	0	0	0	189	0	0	2	75.60
C23	6	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	1	1	1	0	0	234	0	0	93.60
C24	0	0	0	5	1	6	1	1	1	0	0	1	0	0	3	0	0	2	22	0	0	0	0	207	0	82.80
C25	1	1	2	1	0	1	0	0	0	3	1	0	0	0	1	0	0	0	0	4	0	1	0	0	234	93.60
P(%)	76.30	84.17	65.50	96.30	77.99	86.67	92.40	88.66	93.87	94.25	90.31	91.73	90.46	81.85	91.44	83.02	81.75	78.90	90.33	96.94	78.35	75.00	97.50	90.00	95.12	**86.88**

Table 8. Confusion matrix of classification results based on VDCC64.

Class	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20	C21	C22	C23	C24	C25	P(%)
C1	216	4	1	0	0	0	4	0	1	2	0	0	0	0	1	1	0	6	0	1	8	2	1	1	1	86.40
C2	8	205	3	1	2	0	1	1	0	0	1	1	1	1	4	2	3	6	0	0	6	2	0	2	0	82.00
C3	1	1	158	0	10	0	1	1	0	0	1	0	0	4	0	22	4	4	0	0	3	38	0	2	0	63.20
C4	3	4	0	234	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	5	0	93.60
C5	1	0	5	0	207	1	0	2	0	0	0	0	0	12	0	1	14	3	0	0	4	0	0	0	0	82.80
C6	1	0	1	0	1	220	0	0	0	0	0	0	0	10	3	0	5	1	0	0	3	1	0	2	2	88.00
C7	7	0	0	0	0	0	227	0	0	0	0	0	1	0	0	1	0	8	0	0	1	1	0	4	0	90.80
C8	0	0	2	0	7	1	0	219	0	0	3	0	2	6	0	0	0	7	0	0	2	1	0	0	0	87.60
C9	1	0	0	0	0	0	0	0	246	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	98.40
C10	1	0	0	0	0	0	0	0	0	248	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	99.20
C11	2	0	2	0	1	1	0	1	0	0	230	0	6	1	1	0	0	2	0	0	2	1	0	0	0	92.00
C12	0	0	1	0	0	1	0	0	0	0	0	245	0	0	1	0	0	0	0	0	2	0	0	0	0	98.00
C13	2	0	0	0	0	0	0	1	0	0	8	0	239	0	0	0	0	0	0	0	0	0	0	0	0	95.60
C14	0	0	3	0	10	4	0	0	0	0	0	1	0	222	1	1	1	1	0	0	2	0	0	1	3	88.80
C15	0	5	0	0	0	2	0	0	0	0	0	0	0	0	238	0	0	0	0	0	3	0	0	2	0	95.20
C16	0	0	35	0	2	1	0	0	0	0	0	0	0	0	0	206	0	2	0	0	4	0	0	0	0	82.40
C17	0	0	2	0	13	1	0	1	0	0	0	0	0	2	0	0	224	0	0	1	2	4	0	0	0	89.60
C18	6	5	4	0	7	1	4	4	4	0	3	0	2	0	0	1	3	202	0	0	2	2	0	0	0	80.80
C19	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	239	0	0	0	1	8	0	95.60
C20	8	2	0	0	1	0	0	4	1	2	1	0	1	0	0	0	2	0	0	225	3	0	0	0	0	90.00
C21	5	5	11	0	7	0	0	1	0	0	0	4	0	8	1	3	2	1	0	0	196	3	0	3	0	78.40
C22	3	0	26	0	7	0	0	6	0	0	0	0	0	1	0	1	4	3	0	0	4	194	0	1	0	77.60
C23	10	0	0	1	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	4	0	0	230	1	0	92.00
C24	0	1	1	2	2	5	1	1	0	0	0	0	0	1	6	0	0	3	11	0	0	0	0	216	0	86.40
C25	1	2	1	0	0	2	1	0	0	1	0	0	1	1	0	0	0	1	0	1	1	0	0	0	237	94.80
P(%)	78.26	87.23	61.72	97.91	74.73	91.67	94.58	90.50	96.09	98.02	93.12	97.61	94.47	82.53	92.61	86.19	85.17	80.16	95.60	96.57	79.03	77.29	99.14	87.10	97.53	**88.37**

4.3.2. Retrieval Performance of the DPCA

In this section, we show the retrieval performance achieved using the DPCA scheme described in Section 3. Tables 9 and 10 show the top-100 and top-150 retrieval precision results for each class, respectively. The results show that when we compress the original deep feature codes to dimensions of 1024, 256 and 64, only a few classes show improved retrieval precision. The 32-dimensional features show better retrieval performance for certain classes. The wharf class shows the greatest increase in precision when the original deep features are compressed via the PCA method. Specifically, at the P@100 level, an 8.45% precision improvement is achieved by compressing the deep features from the Alexnet model and a 17.57% precision improvement is achieved by compressing the deep features from the VGG model, whereas at the P@150 level, 8.75% and 18.33% precision improvements are achieved by compressing the deep features from the Alexnet and VGG models, respectively. However, Tables 9 and 10 show that the improvements in retrieval performance are limited compared with the results based on the original deep features. Many of the best retrieval results are obtained using the original deep features. In other words, the retrieval performance decreases for certain classes when extracting the feature codes via the PCA method. This finding indicates that certain important feature information can be lost while compressing the deep features to lower dimensionalities. A comparison of the information in Tables 1 and 2 shows that our DCC method can learn shorter feature codes and achieve much better retrieval performance.

Table 9. P@100 values obtained using the DPCA method.

Class	Alexnet	APCA1024	APCA256	APCA64	APCA32	VGG	VPCA1024	VPCA256	VPCA64	VPCA32
agricultural	0.5241	0.3437	0.4780	0.5340	**0.5750**	**0.6908**	0.5730	0.6571	0.6785	0.6765
airport	0.5979	0.3119	0.4946	0.6025	**0.6458**	0.5594	0.2680	0.4884	0.6046	**0.6392**
basketball court	0.4384	0.3179	0.4178	0.4554	**0.4560**	0.3478	0.1729	0.3282	0.3878	**0.3933**
bridge	0.8344	0.4913	0.7283	0.8194	**0.8567**	0.7831	0.4218	0.7400	0.8179	**0.8358**
building	0.5965	0.3791	0.4954	0.5677	**0.6136**	0.5119	0.1907	0.4404	0.5844	**0.6321**
container	0.6688	0.4536	0.5662	0.6451	**0.6834**	0.7718	0.3769	0.6794	0.7577	**0.7841**
fishpond	**0.8289**	0.5668	0.7296	0.7932	0.8191	0.8160	0.4713	0.7211	0.8155	**0.8306**
footbridge	0.6231	0.4420	0.5352	0.5994	**0.6249**	0.7177	0.3492	0.6279	0.7328	**0.7601**
forest	**0.9126**	0.8978	0.8862	0.8953	0.9056	**0.9407**	0.9205	0.9302	0.9331	0.9355
greenhouse	**0.9234**	0.7992	0.8624	0.8991	0.9143	**0.9458**	0.7723	0.8988	0.9350	0.9413
intersection	0.6707	0.4488	0.5590	0.6351	**0.6770**	0.8196	0.5124	0.7345	0.7922	**0.8227**
oiltank	**0.8043**	0.2754	0.5695	0.7278	0.7807	**0.8223**	0.0632	0.2601	0.6088	0.7353
overpass	0.6624	0.3990	0.5523	0.6520	**0.6898**	0.7768	0.2972	0.6504	0.7757	**0.8046**
parking lot	0.5376	0.3586	0.4531	0.5247	**0.5584**	0.5711	0.1822	0.4463	0.5614	**0.6066**
plane	0.8502	0.6412	0.7668	0.8351	**0.8570**	0.7719	0.3859	0.6749	0.7898	**0.8289**
playground	**0.7081**	0.4823	0.6079	0.6666	0.6985	**0.7097**	0.4133	0.6124	0.6761	0.6987
residential	**0.7020**	0.5557	0.6076	0.6549	0.6856	**0.7307**	0.3774	0.6228	0.7034	0.7286
river	0.4669	0.3353	0.4243	0.4892	**0.5114**	0.5115	0.3201	0.4739	0.5595	**0.5846**
ship	**0.8885**	0.7902	0.8394	0.8644	0.8720	**0.9245**	0.7502	0.8822	0.9097	0.9076
solar power area	**0.7502**	0.4966	0.6166	0.6762	0.7150	**0.8157**	0.5932	0.7326	0.7819	0.8028
square	0.5082	0.3472	0.4348	0.4917	**0.5215**	0.5153	0.3094	0.4677	0.5301	**0.5608**
tenniscourt	0.4515	0.2690	0.4001	0.4710	**0.4841**	0.4915	0.2042	0.4101	0.5293	**0.5685**
water	**0.8726**	0.8764	0.8492	0.8595	0.8652	0.9249	**0.9288**	0.9056	0.9046	0.9100
wharf	0.6476	0.3556	0.5835	0.6725	**0.7321**	0.5102	0.1989	0.4853	0.6164	**0.6859**
workshop	**0.8676**	0.6754	0.7760	0.8182	0.8486	**0.8378**	0.4671	0.6946	0.7868	0.8275

Tables 11 and 12 show the comprehensive retrieval accuracies for the different dimensional features for the entire dataset. As shown in Tables 11 and 12, sharp decreases in the comprehensive retrieval performance are observed when compressing the original deep features to different dimensional features. Note that the reduction in precision declines as the dimensionality is reduced. When deep features are compressed to a dimensionality of 32, retrieval precision improvements occur; this finding corresponds to the information presented in Tables 9 and 10. Based on this observation, we also calculate the retrieval accuracies achieved when the deep features are compressed to a dimensionality of 16. Specifically, at the P@100 level, 16-dimensional features compressed from the deep features of the Alexnet framework achieve a retrieval precision of 69.05%, which is 0.30% lower

than the precision of the original deep features. With regard to the deep features of the VGG framework, the compressed 16-dimensional features achieve a retrieval precision of 73.01%, which is 1.73% higher than the precision of the original deep features. The P@150 level generally shows the same results. Nevertheless, all these results show reduced performance compared with that of the compressed 32-dimensional features, as shown in Tables 11 and 12. A comparison of our DCC method, which can achieve significant improvements in the retrieval performance for all the different features at lower dimensionalities, with the DPCA method indicates that the DPCA method only achieves limited improvements in retrieval performance for the 32- and 16-dimensional features. This phenomenon mainly occurs because the PCA method is a linear compression scheme, while the features of remote sensing images should have non-linear relations. This finding indicates that our proposed DCC method is an efficient method for learning more powerful image feature representations with lower dimensionalities. Figure 3 shows a query image of a wharf and the corresponding top-5 irrelevant retrieval images obtained using the Alexnet based DCC and DPCA methods. The results show that the orders of irrelevant images obtained using the ADCC32, ADCC64 and ADCC256 models are much greater than those obtained using other models at the same position, which indicates that our proposed DCC method has a better retrieval performance. Specifically, ADCC64 shows the best results, which is consistent with the results shown in Tables 1–4 and prior analyses.

Table 10. P@150 values obtained using the DPCA method.

Class	Alexnet	APCA1024	APCA256	APCA64	APCA32	VGG	VPCA1024	VPCA256	VPCA64	VPCA32
agricultural	0.4722	0.2952	0.4206	0.4775	**0.5231**	**0.6420**	0.4801	0.5987	0.6263	0.6277
airport	0.5531	0.2550	0.4313	0.5447	**0.5949**	0.5018	0.2087	0.4139	0.5462	**0.5854**
basketball court	0.4172	0.2780	0.3827	0.4297	**0.4334**	0.3246	0.1432	0.2972	0.3653	**0.3781**
bridge	0.8001	0.3882	0.6507	0.7722	**0.8273**	0.7309	0.3183	0.6610	0.7765	**0.8032**
building	0.5496	0.3202	0.4345	0.5134	**0.5689**	0.4697	0.1531	0.3858	0.5366	**0.5924**
container	0.6094	0.3606	0.4845	0.5689	**0.6143**	0.7127	0.2811	0.5971	0.7000	**0.7301**
fishpond	**0.7986**	0.4473	0.6573	0.7500	0.7867	0.7817	0.3718	0.6474	0.7741	**0.7951**
footbridge	0.5793	0.3772	0.4769	0.5515	**0.5804**	0.6777	0.2611	0.5488	0.6856	**0.7181**
forest	**0.8868**	0.8422	0.8435	0.8596	0.8797	**0.9281**	0.8858	0.9099	0.9202	0.9237
greenhouse	**0.9032**	0.7173	0.8151	0.8660	0.8889	**0.9329**	0.6545	0.8552	0.9178	0.9254
intersection	0.6257	0.3729	0.4929	0.5817	**0.6287**	0.7790	0.3952	0.6619	0.7459	**0.7837**
oiltank	**0.7667**	0.2063	0.4497	0.6534	0.7235	**0.7755**	0.0471	0.1905	0.4880	0.6449
overpass	0.6150	0.3333	0.4873	0.5950	**0.6419**	0.7295	0.2230	0.5583	0.7171	**0.7628**
parking lot	0.4980	0.3011	0.4029	0.4829	**0.5147**	0.5241	0.1376	0.3714	0.4987	**0.5514**
plane	0.8074	0.5406	0.7004	0.7856	**0.8160**	0.7107	0.2869	0.5809	0.7274	**0.7809**
playground	**0.6707**	0.3921	0.5415	0.6194	0.6614	**0.6617**	0.3164	0.5398	0.6258	0.6567
residential	**0.6578**	0.4598	0.5402	0.6029	0.6433	0.6838	0.2900	0.5482	0.6502	**0.6872**
river	0.4204	0.2916	0.3747	0.4364	**0.4576**	0.4581	0.2604	0.4137	0.5053	**0.5309**
ship	**0.8631**	0.6881	0.7943	0.8384	0.8515	**0.8934**	0.6171	0.8269	0.8890	0.8893
solararea	**0.7139**	0.3878	0.5351	0.6172	0.6637	**0.7685**	0.4588	0.6490	0.7236	0.7558
square	0.4662	0.2961	0.3861	0.4442	**0.4769**	0.4580	0.2542	0.4050	0.4697	**0.5042**
tennis court	0.4130	0.2317	0.3587	0.4279	**0.4411**	0.4426	0.1625	0.3542	0.4799	**0.5250**
water	**0.8554**	0.8458	0.8145	0.8366	0.8446	**0.9235**	0.9274	0.8986	0.9012	0.9072
wharf	0.6024	0.2947	0.5113	0.6191	**0.6899**	0.4549	0.1540	0.4114	0.5547	**0.6382**
workshop	**0.8387**	0.5634	0.7200	0.7777	0.8155	**0.7969**	0.3512	0.5970	0.7257	0.7836

Table 11. Retrieval accuracies of the different dimensional features obtained using the DPCA method according to the P@100 values.

Model	Accuracy	Model	Accuracy
Alexnet	69.35%	VGG	71.27%
APCA1024	49.24%	VPCA1024	42.08%
APCA256	60.94%	VPCA256	62.26%
APCA64	67.40%	VPCA64	71.09%
APCA32	**70.37%**	VPCA32	**74.01%**

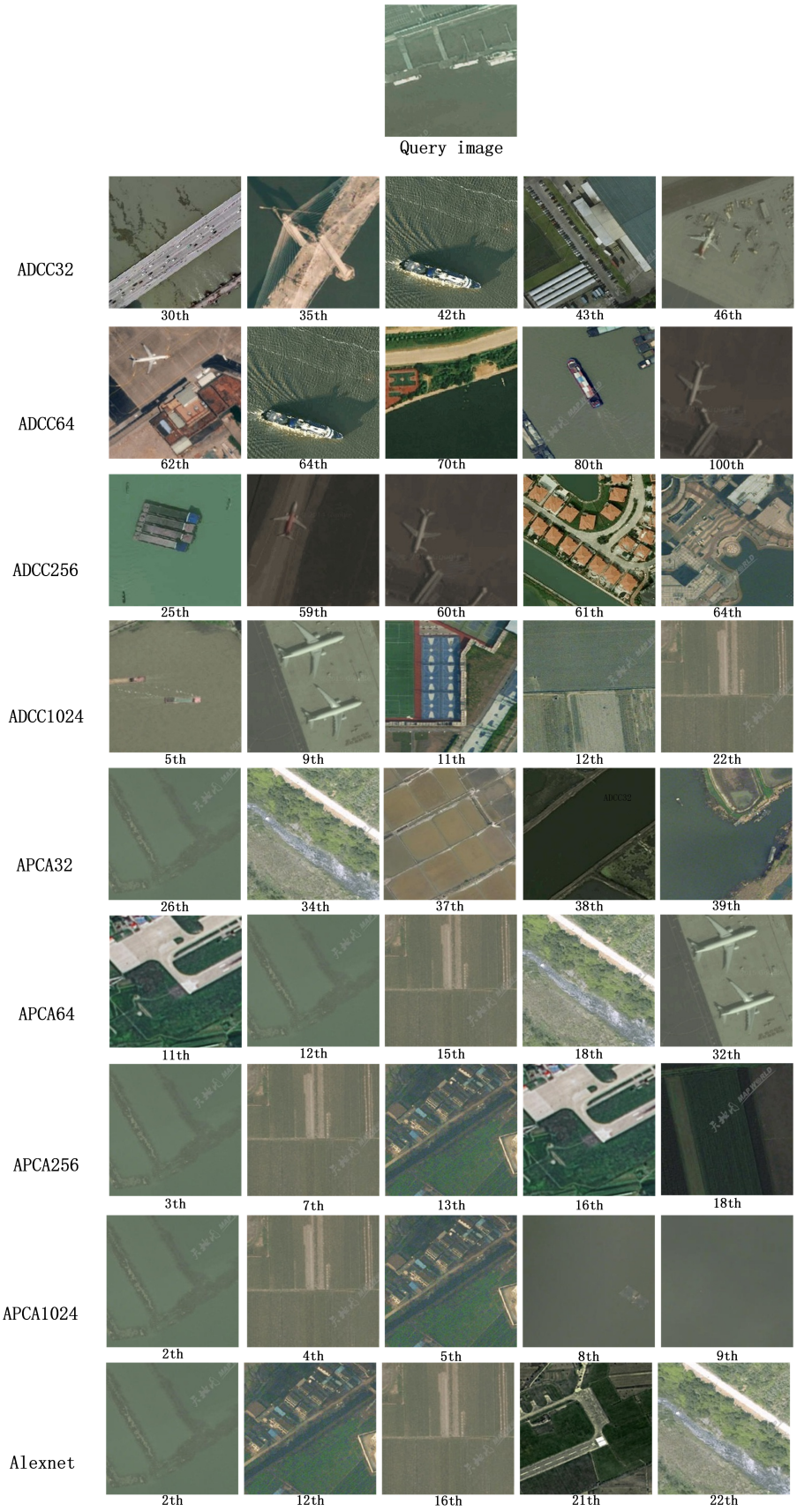

Figure 3. Top-5 retrieval results that are irrelevant to the query image.

Table 12. Retrieval accuracies of the different dimensional features obtained using the DPCA method according to the P@150 values.

Model	Accuracy	Model	Accuracy
Alexnet	65.54%	VGG	67.05%
APCA1024	41.95%	VPCA1024	34.56%
APCA256	54.83%	VPCA256	55.69%
APCA64	62.61%	VPCA64	66.20%
APCA32	**66.27**%	VPCA32	**69.92**%

4.3.3. Comparison

In this section, we compare the performances of the DCC and DPCA methods. The mAP results for the evaluated methods are listed in Table 13, which shows that all DCC frameworks substantially outperform the baseline CNN frameworks. Compared with the baseline frameworks, the 64-dimensional features extracted by our proposed DCC method based on the Alexnet or VGG frameworks achieve the best results. Specifically, absolute mAP increases of 8.51% and 8.64% are observed for the 64-dimensional features using the Alexnet- and VGG-based DCC frameworks, respectively. For the DPCA method, only 32-dimensional features compressed from the deep features of the VGG framework achieve better mAP performance compared with the baseline framework. However, only the VPCA32 features can improve the performance of the image retrieval. In contrast, the 32-dimensional features extracted by our proposed DCC method can highly outperform those of the DPCA method. Furthermore, the mAP values of all DCC frameworks are greater than those of the DPCA method, as shown in Table 13.

Table 13. Comparison of the retrieval mAP values of the DCC and DPCA methods.

Model	mAP	Model	mAP	Model	mAP	Model	mAP
Alexnet	0.5779	VGG	0.5795	-	-	-	-
ADCC1024	0.6257	VDCC1024	0.6203	APCA1024	0.3108	VPCA1024	0.2608
ADCC256	0.6615	VDCC256	0.6524	APCA256	0.4258	VPCA256	0.4311
ADCC64	**0.6630**	VDCC64	**0.6659**	APCA64	0.5224	VPCA64	0.5512
ADCC32	0.6475	VDCC32	0.6030	APCA32	**0.5714**	VPCA32	**0.5971**

The recall-precision curves of the different DCC and DPCA methods are plotted in Figure 4. As shown, low-dimensional features obtained using our proposed DCC method outperform those of the baseline frameworks (Figure 4a,b), which is also consistent with Tables 3 and 13. Specifically, for Alexnet-based frameworks, 64-dimensional features have the best performance, with high recall and precision retrieval results (Figure 4a). Note that the 32-dimensional DCC features based on Alexnet achieve much better results at lower recall levels than the corresponding baseline features, which is very desirable for precision-oriented image retrieval. For high recall levels, the 256-dimensional features have the best performance. Therefore, this amount is suitable for recall-oriented image retrieval. For VGG-based frameworks, the 64-dimensional features also outperform all the other dimensional features and show comparative results at high recall levels, even when compared with the 256-dimensional features (Figure 4b). For the DPCA method, only 32-dimensional features show comparative results when compared with the original deep features (Figure 4c,d). In general, the 64-dimensional features based on our proposed DCC frameworks show the best results, and they dramatically improve the retrieval performance.

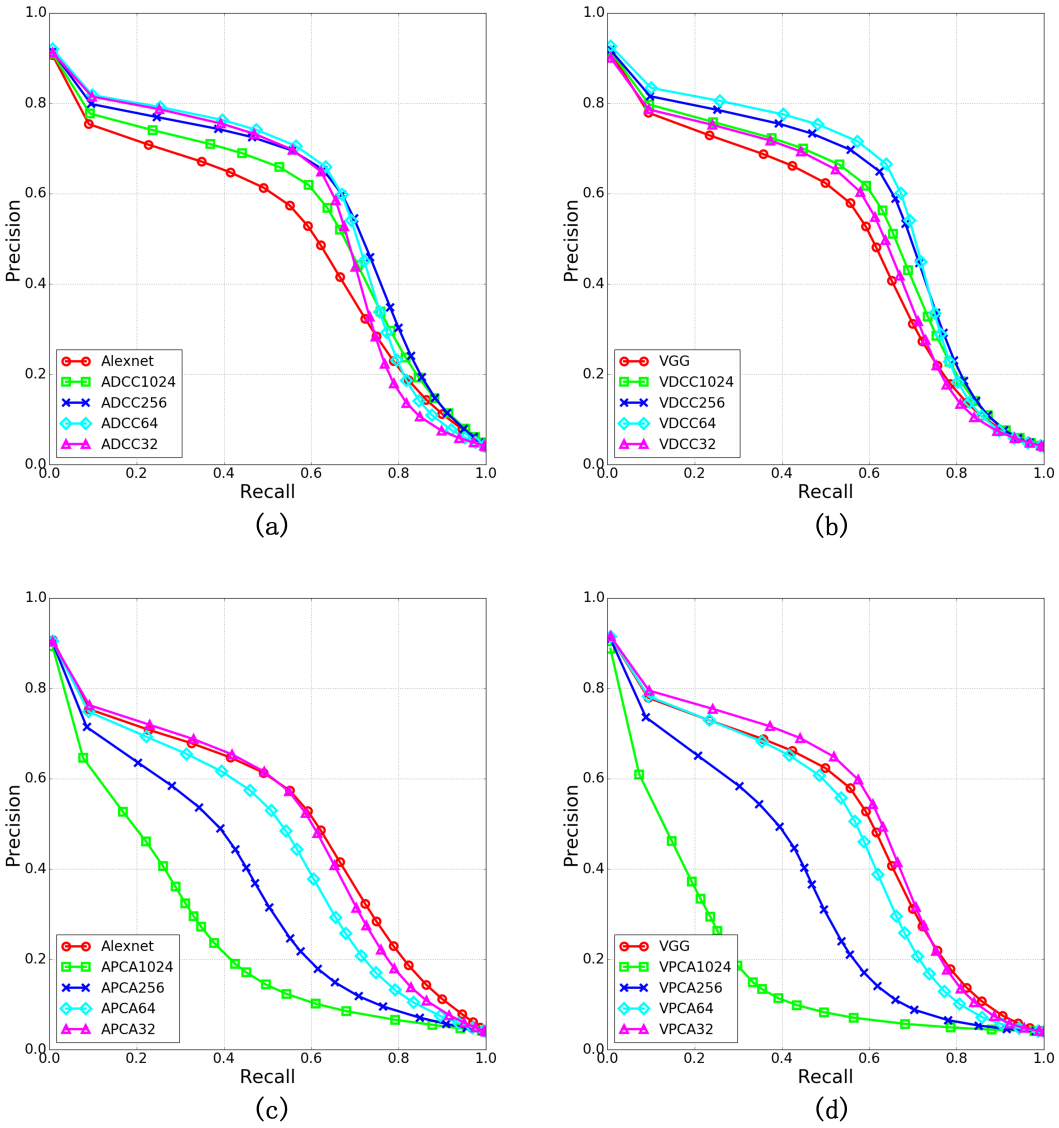

Figure 4. Recall-precision curves for the different methods. (**a**) DCC method based on Alexnet; (**b**) DCC method based on VGG; (**c**) DPCA method based on Alexnet; and (**d**) DPCA method based on VGG.

A further comparison based on the above analyses is performed by plotting the best results of each framework in Figure 5. The results reveal that the 64-dimensional features extracted by our DCC frameworks significantly outperform those of the baseline frameworks and the DPCA method, which demonstrates the effectiveness and practicability of our proposed DCC method for remote sensing image retrieval. Specifically, the findings indicate that the 64-dimensional DCC features based on the Alexnet and the VGG frameworks generally have the same performance. Note that Alexnet-based DCC models are shallow frameworks, whereas VGG-based models are much deeper. This comparison shows that our proposed DCC method can use a shallow CNN framework to realize a performance comparable to that achieved using a much deeper CNN framework. Furthermore, the 64-dimensional DCC features based on the Alexnet framework achieve much better retrieval results than the original VGG framework, which is also consistent with the results shown in Tables 3 and 13. This finding reveals that our proposed DCC method can greatly improve upon the performance of a shallow CNN framework and can be used to obtain a greater precision than that achieved using a deeper CNN framework, which shows the advantages of improving storage and efficiency simultaneously.

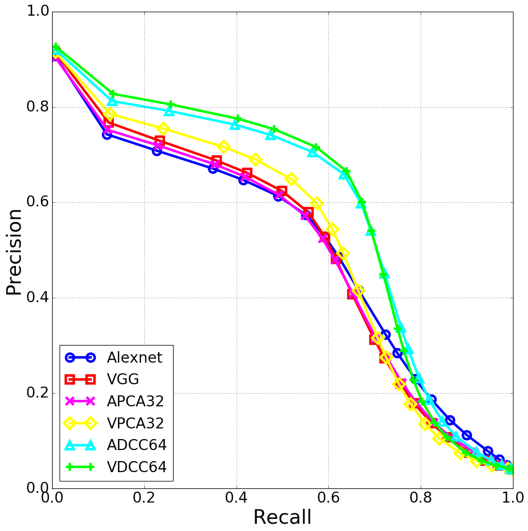

Figure 5. Different methods with the best performance.

5. Visual Understanding of the DCC

The FWM can help us better understand the nature of DCC features via visualization. Based on previous results and analyses, the ADCC64 and VDCC64 models show the best retrieval performances, which demonstrates that the 64-dimensional DCC features can represent an image more appropriately than the 4096-dimensional features extracted from traditional DCNN frameworks. Therefore, the ADCC64 and VDCC64 models are selected for the weighted feature visualization. We also show the FWM based on the traditional Alexnet and VGG DCNN frameworks for comparison. Specifically, 10 samples of 25 classes from the dataset are randomly selected for visualization. The results of the FWMs are shown in Figure 6, which indicates that our proposed FWM method can successfully illustrate the regions of objects in an image. Red regions may include intersections with objects in the images, which demonstrates the efficiency of our proposed FWM visualization method. Simultaneously, red regions in the FWMs indicate the features that are focused on by the DCNN frameworks; this information is preserved in the final deep features. This arrangement helps explain why the learned deep features provide accurate representations of the images.

A comparison of the FWMs based on the ADCC64 and VDCC64 models with those based on the baseline CNN frameworks shows that the former frameworks could indicate the localization and extent of objects more precisely in the images for certain classes, such as playground, parking lot and tennis court. However, for the FWMs based on the original Alexnet and VGG frameworks, a greater amount of information is scattered in a messy manner, which could represent the noisy information of objects. This information can also be learned and preserved in the final feature codes; thus, it may have a negative influence on the retrieval performance. This circumstance helps us intuitively understand how our proposed DCC frameworks can significantly outperform the traditional Alexnet and VGG frameworks for these classes. This approach also corresponds to the retrieval precision results shown in Table 1. Although noisy information is shown in the FWM of the footbridge class, our proposed DCC methods tend to focus more precisely on the position and region of the footbridge. For the building class, the DCC frameworks tend to learn more aggregated information from the building top, which is much different from the scattered information found by the original frameworks, especially for the VGG model. Similarly, the VDCC64 model remarkably improves the retrieval performance of the building class by 17.10% compared with the original VGG model. For the plane class, the DCC method can learn more information from different parts of a plane. Regarding the FWM of the oil tank, the DCC and traditional frameworks can distinguish all the oil tanks in the image. Note that the VDCC64 model has better results compared with the VGG model. For the airport and bridge classes, all CNN frameworks tend to learn the background information of the objects rather than the airport runways

or bridge bodies. This tendency means that the CNN frameworks attempt to discern the objects from their backgrounds. The DCNN frameworks do not always learn object information, which is similar to human cognition. Additionally, the background information is vital to class discrimination information learning. Based on these observations, the proposed FWM visualization method based on our DCC frameworks should also be applicable to class discrimination, scene recognition and other non-classification tasks.

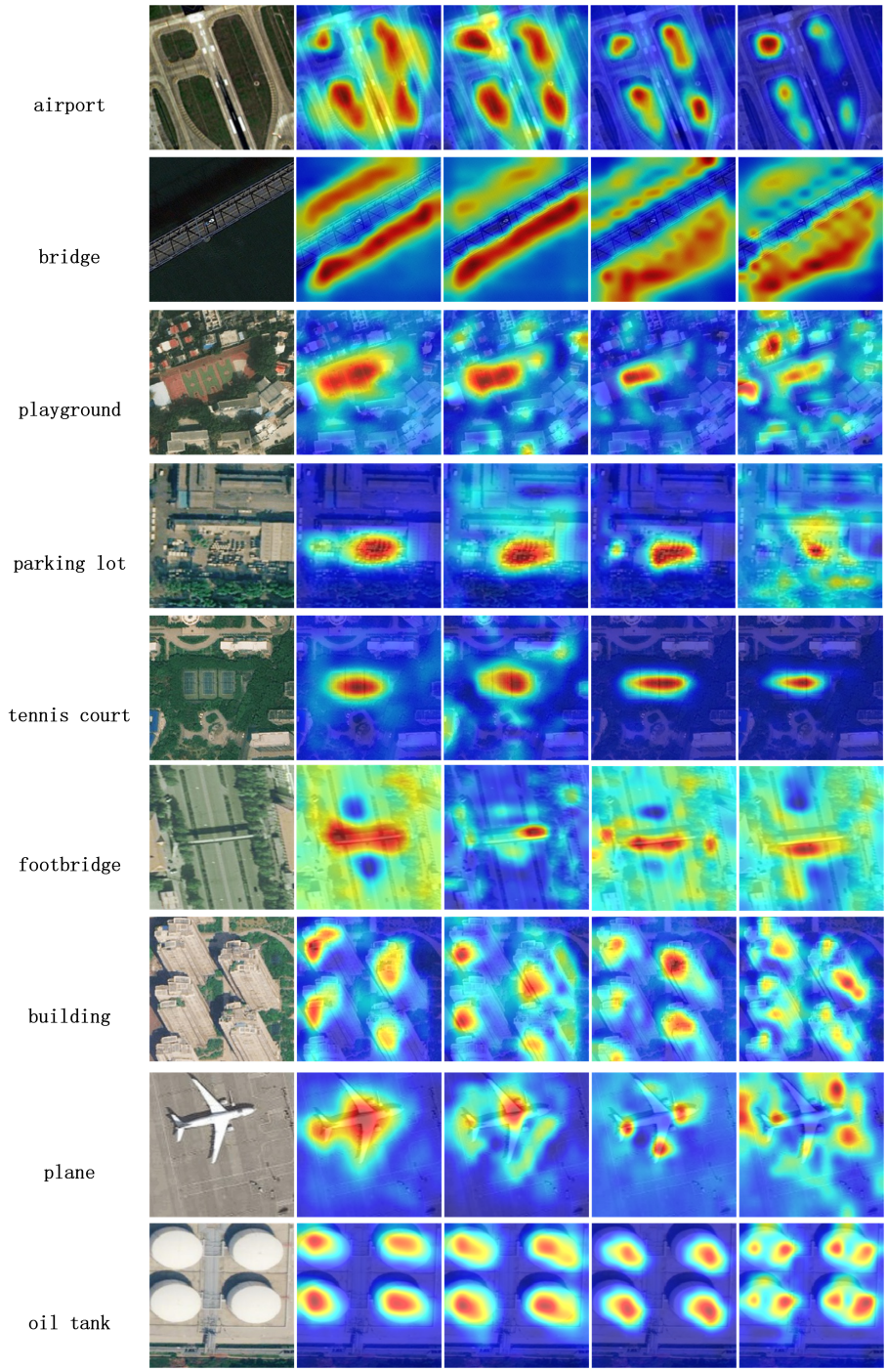

Figure 6. Feature Weighted Maps of the DCC models. The first column of each row is the original image. The second to the fourth columns are the FWMs extracted by the ADCC64, Alexnet, VDCC64 and VGG models, respectively.

6. Conclusions

In this work, we propose learning DCCs for CB-HRRS-IR. Extensive experiments are conducted to learn feature codes with dimensionalities extending from tens to thousands for image retrieval based on a large-scale remote sensing image archive. A PCA is also employed to compress the deep features. The experimental results reveal that our proposed DCC method can remarkably outperform traditional DCNN frameworks and the DPCA method. Additionally, the 64-dimensional DCC features yield the best retrieval results. To further understand the learned deep features, we explore a feature-oriented visualization method FWM, which demonstrates that our proposed DCC method can learn more powerful information for image representation. This feature-oriented visualization method can also be generalized to any CNN framework and generate FWMs from any layer for a classification-oriented or non-classification task.

Acknowledgments: This research is supported by the National Key Research and Development Program of China (No. 2016YFB0502602), Internally Funded Projects by LIESMARS and Fundamental Research Funds for the Central Universities.

Author Contributions: Z.X., Y.L. and D.L. conceived and designed the experiments; Y.L. performed the experiments, analyzed the data and wrote the paper; C.W., G.T. and J.L. contributed materials.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, Y.; Newsam, S. Geographic image retrieval using local invariant features. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 818–832.
2. Li, Y.; Zhang, Y.; Tao, C.; Zhu, H. Content-Based High-Resolution Remote Sensing Image Retrieval via Unsupervised Feature Learning and Collaborative Affinity Metric Fusion. *Remote Sens.* **2016**, *8*, 709.
3. Demir, B.; Bruzzone, L. Hashing-Based Scalable Remote Sensing Image Search and Retrieval in Large Archives. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 892–904.
4. Bretschneider, T.; Cavet, R.; Kao, O. Retrieval of remotely sensed imagery using spectral information content. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Toronto, ON, Canada, 24–28 June 2002; Volume 4, pp. 2253–2255.
5. Geng, W.; Zhang, J.; Zhuo, L.; Liu, J.; Chen, L. Creating Spectral Words for Large-Scale Hyperspectral Remote Sensing Image Retrieval. *Pacific Rim Conference on Multimedia*; Springer: New York, NY, USA, 2016; pp. 116–125.
6. Hongyu, Y.; Bicheng, L.; Wen, C. Remote sensing imagery retrieval based-on Gabor texture feature classification. In Proceedings of the 7th IEEE International Conference on Signal, Beijing, China, 31 August–4 September 2004; Volume 1, pp. 733–736.
7. Aptoula, E. Remote sensing image retrieval with global morphological texture descriptors. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 3023–3034.
8. Yang, F.P.; Hao, M.L. Effective Image Retrieval Using Texture Elements and Color Fuzzy Correlogram. *Information* **2017**, *8*, 27.
9. Ma, A.; Sethi, I.K. Local shape association based retrieval of infrared satellite images. In Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05), Irvine, CA, USA, 12–4 December 2005; pp. 551–557.
10. Scott, G.J.; Klaric, M.N.; Davis, C.H.; Shyu, C.R. Entropy-balanced bitmap tree for shape-based object retrieval from large-scale satellite imagery databases. *IEEE Trans. Geosci. Remote Sens.* **2011**, *49*, 1603–1616.
11. Yang, J.; Liu, J.; Dai, Q. An improved Bag-of-Words framework for remote sensing image retrieval in large-scale image databases. *Int. J. Digit. Earth* **2015**, *8*, 273–292.
12. Wang, Y.; Zhang, L.; Tong, X.; Zhang, L.; Zhang, Z.; Liu, H.; Xing, X.; Mathiopoulos, P.T. A three-layered graph-based learning approach for remote sensing image retrieval. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 6020–6034.
13. Bosilj, P.; Aptoula, E.; Lefèvre, S.; Kijak, E. Retrieval of Remote Sensing Images with Pattern Spectra Descriptors. *ISPRS Int. J. Geo-Inf.* **2016**, *5*, 228.

14. Pham, M.T.; Mercier, G.; Regniers, O.; Michel, J. Texture Retrieval from VHR Optical Remote Sensed Images Using the Local Extrema Descriptor with Application to Vineyard Parcel Detection. *Remote Sens.* **2016**, *8*, 368.
15. Du, Z.; Li, X.; Lu, X. Local Structure Learning in High Resolution Remote Sensing Image Retrieval. *Neurocomputing* **2016**, *207*, 813–822.
16. Zeng, Z. A Novel Local Structure Descriptor for Color Image Retrieval. *Information* **2016**, *7*, 9.
17. Liu, T.; Zhang, L.; Li, P.; Lin, H. Remotely sensed image retrieval based on region-level semantic mining. *EURASIP J. Image Video Process.* **2012**, *2012*, 4.
18. Wang, M.; Song, T. Remote Sensing Image Retrieval by Scene Semantic Matching. *IEEE Trans. Geosci. Remote Sens.* **2013**, *51*, 2874–2886.
19. John, L.M.; Bhandari, K.A. A Novel Method for Satellite Image Retrieval using Semantic Mining and Hashing. *Int. J. Comput. Appl.* **2016**, *147*, 33–36.
20. Hinton, G.; Deng, L.; Yu, D.; Dahl, G.E.; Mohamed, A.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T.N. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Process. Mag.* **2012**, *29*, 82–97.
21. Yu, D.; Seltzer, M.L.; Li, J.; Huang, J.T.; Seide, F. Feature Learning in Deep Neural Networks—Studies on Speech Recognition Tasks. *arXiv* **2013**, arXiv:1301.3605.
22. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. In Proceedings of the International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
23. Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks. *arXiv* **2013**, arXiv:1311.2901v3.
24. Cheng, G.; Zhou, P.; Han, J. Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 7405–7415.
25. Long, Y.; Gong, Y.; Xiao, Z.; Liu, Q. Accurate Object Localization in Remote Sensing Images Based on Convolutional Neural Networks. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 2486–2498.
26. Huang, E.H.; Socher, R.; Manning, C.D.; Ng, A.Y. Improving word representations via global context and multiple word prototypes. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers, Jeju Island, Korea, 8–14 July 2012; Volume 1, pp. 873–882.
27. Mikolov, T.; Yih, W.T.; Zweig, G. Linguistic Regularities in Continuous Space Word Representations. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Atlanta, GA, USA, 9–15 June 2013; Volume 13, pp. 746–751.
28. Zhou, W.; Shao, Z.; Diao, C.; Cheng, Q. High-resolution remote-sensing imagery retrieval using sparse features by auto-encoder. *Remote Sens. Lett.* **2015**, *6*, 775–783.
29. Napoletano, P. Visual descriptors for content-based retrieval of remote sensing images. *arXiv* **2016**, arXiv:1602.00970.
30. Zhou, W.; Newsam, S.; Li, C.; Shao, Z. Learning Low Dimensional Convolutional Neural Networks for High-Resolution Remote Sensing Image Retrieval. *Remote Sens.* **2017**, *9*, 489.
31. Springenberg, J.T.; Dosovitskiy, A.; Brox, T.; Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv* **2014**, arXiv:1412.6806 .
32. Mahendran, A.; Vedaldi, A. Understanding deep image representations by inverting them. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5188–5196.
33. Dosovitskiy, A.; Brox, T. Inverting visual representations with convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 27–30 June 2016; pp. 4829–4837.
34. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning deep features for discriminative localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 27–30 June 2016, pp. 2921–2929.
35. Selvaraju, R.R.; Das, A.; Vedantam, R.; Cogswell, M.; Parikh, D.; Batra, D. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv* **2016**, arXiv:1610.02391 .
36. Xia, G.S.; Yang, W.; Delon, J.; Gousseau, Y.; Sun, H.; Maître, H. Structural High-resolution Satellite Image Indexing. In Proceedings of the ISPRS TC VII Symposium—100 Years ISPRS, Vienna, Austria, 5–7 July 2010, pp. 298–303.

37. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of the ICML 2013: The 30th International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; Volume 30.
38. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE International Conference on Computer Vision, Boston, MA, USA, 7–12 June 2015; pp. 1026–1034.
39. Hinton, G.E.; Salakhutdinov, R.R. Reducing the Dimensionality of Data with Neural Networks. *Science* **2006**, *313*, 504.
40. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.