remote sensing

MDPI

Article

Fast Binary Coding for the Scene Classification of High-Resolution Remote Sensing Imagery

Fan Hu [1,2], Gui-Song Xia [1,*], Jingwen Hu [1,2], Yanfei Zhong [1] and Kan Xu [3]

[1] State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing (LIESMARS), Wuhan 430079, China; hfmelizabeth@gmail.com (F.H.); hujingwen@whu.edu.cn (J.H.); zhongyanfei@whu.edu.cn (Y.Z.)

[2] Electronic Information School, Wuhan University, Wuhan 430072, China

[3] Global Navigation Satellite System (GNSS) Research Center, Wuhan University, Wuhan 430079, China; jcarloswhu@msn.com

* Correspondence: guisong.xia@whu.edu.cn; Tel.: +86-27-6877-9908

Academic Editors: Guoqing Zhou and Prasad S. Thenkabail
Received: 21 January 2016; Accepted: 22 June 2016; Published: 30 June 2016

Abstract: Scene classification of high-resolution remote sensing (HRRS) imagery is an important task in the intelligent processing of remote sensing images and has attracted much attention in recent years. Although the existing scene classification methods, e.g., the bag-of-words (BOW) model and its variants, can achieve acceptable performance, these approaches strongly rely on the extraction of local features and the complicated coding strategy, which are usually time consuming and demand much expert effort. In this paper, we propose a fast binary coding (FBC) method, to effectively generate efficient discriminative scene representations of HRRS images. The main idea is inspired by the unsupervised feature learning technique and the binary feature descriptions. More precisely, equipped with the unsupervised feature learning technique, we first learn a set of optimal "filters" from large quantities of randomly-sampled image patches and then obtain feature maps by convolving the image scene with the learned filters. After binarizing the feature maps, we perform a simple hashing step to convert the binary-valued feature map to the integer-valued feature map. Finally, statistical histograms computed on the integer-valued feature map are used as global feature representations of the scenes of HRRS images, similar to the conventional BOW model. The analysis of the algorithm complexity and experiments on HRRS image datasets demonstrate that, in contrast with existing scene classification approaches, the proposed FBC has much faster computational speed and achieves comparable classification performance. In addition, we also propose two extensions to FBC, i.e., the spatial co-occurrence matrix and different visual saliency maps, for further improving its final classification accuracy.

Keywords: scene classification; filter banks; feature representation; binary coding; high-resolution remote sensing images

1. Introduction

In recent years, an increasing number of commercial satellite sensors of high resolution have been successfully launched, and a new era of "big data" for remote sensing is coming [1,2]. The more and more mature remote sensing imaging technologies have made massive raw high-resolution (HR) satellite and aerial image datasets available. Although the high-resolution remotely-sensed (HRRS) images enable us to measure the Earth's surface with more accuracy, the huge volume of images with rich structures has also led to many new problems arising for the intelligent processing of remote sensing data. In the context of "big data", developing fast or even real-time remote sensing image processing systems that are able to greatly enhance work efficiency is now attracting considerable

attention [1–3]. These automatic real-time systems can bring great benefits for many applications that need immediate monitoring and timely feedback, e.g., fire detection, weather forecast and earthquake prediction.

Scene classification of HRRS images is regarded as a fundamental yet challenging task and has attracted much attention in recent years [4–17]. Here, the "scenes" refer to some separated subareas extracted from large satellite images. They usually consist of different types of land covers or objects and possess specific semantic meaning, such as the *residential area*, *industrial area*, *commercial area* and *green land* in a typical urban area satellite image. The scene-based semantic classification plays a significant role in urban planning, land resource management, computer cartography, and many more. The high complexity of spatial and structural patterns in the massive HRRS imagery make the intelligent scene understanding and classification a challenging problem. In order to accurately obtain the scene classes, generating discriminative holistic feature representation for each scene is a key step and highly demanded.

Generally, different scene categories may share some identical thematic classes, which represent a few land-cover types or object classes. For instance, *tree, road and buildings*; these three thematic classes may appear both in the *commercial area* and in the *residential area* at the same time. The bag-of-visual-words (BOW) [18,19] model, which represents each image scene with a histogram where each bin counts the occurrence frequency of codewords (also called visual words) that are formed by vector-quantizing local features using a clustering method (e.g., K-means), is probably the most popular scene classification framework thanks to its simple operation and excellent performance. There are three basic steps in the BOW pipeline for scene classification: extracting local feature descriptors, generating the codebook (composed of all codewords) and encoding local features on the codebook. Among the three steps, feature extraction is the core part and can significantly influence the final classification performance. For the purpose of high classification performance for different image scene datasets, it is crucial to choose or design powerful local feature descriptors; see, e.g., [20–26]. However, designing good features needs too much human effort and expert domain knowledge. Moreover, in the BOW framework, the step of generating the codebook, where the codewords are typically generated by clustering (e.g., K-means) over local features, is usually time consuming. Therefore, a high-efficiency feature coding method is desirable and even an urgent need, especially in the industrial remote sensing scene analysis systems. Nowadays, many binarized feature representation methods [21,27,28] have become increasingly popular, which are very simple and efficient to compute at a fairly fast speed with limited computational resources. Inspired by these binary local features, we develop a global feature representation method for scenes in remotely-sensed images, which integrates the local feature extraction and feature coding stage in an efficient way.

In this paper, we present a fast binary coding scheme for the feature representation of HRRS image scenes. We first randomly sample an amount of local image patches from images in dataset and apply proper unsupervised learning techniques to learn a dictionary, which is regarded as a set of filters. Then, we convolve each image scene with the learned filters and binarize the filter responses according to a predefined threshold. Finally, we convert the binary responses back into a single decimal number and then compute the histogram of the decimal values for each image scene. The final histogram is considered as the holistic feature representation of the image, which can be fed into the classifier for training and testing. In contrast to the typical BOW pipeline, we neither use any hand-crafted features, nor feature encoding techniques, and therefore, greatly improve the computational efficiency. When the set of filters have been generated, the holistic histogram of each image scene can be yielded extremely quickly on common CPUs. Extensive experiments show that we can obtain comparable classification performance at a low computational cost. In addition, to overcome the defects of fast binary coding (FBC), which are disregarding the spatial layout information of images and having much redundancy in histogram representations, we attempt to improve it by introducing the spatial co-occurrence kernel and saliency maps, respectively.

The major contributions of this paper are summarized as follows:

- We develop a fast global feature representation method for image scenes, *named* fast binary coding (FBC), which integrates the local feature extraction and feature coding stage. In the FBC pipeline, we are free of any hand-crafted features, and the features are skillfully learned and encoded in an unsupervised fashion.
- We achieve promising performance with extraordinarily low computational efficiency on scene classification of HRRS images, which can make the FBC an effective and practical method for an industrial scene analysis system.
- On the basis of FBC, we investigate how the spatial kernel extension and various saliency maps can improve the classification performance.

A short version of this paper has appeared in [29]. The remainder of this paper is organized as follows. In Section 2, we briefly review some related works, e.g., BOW-based scene classification methods, binary feature descriptors and unsupervised feature learning. In Section 3, we first introduce the global feature representation of HRRS scenes by FBC and then study various unsupervised algorithms for learning linear filters. In Section 4, two extensions to FBC are presented. The details of our experiments and results are presented in Section 5. Finally, we draw conclusions for this paper with some remarks.

2. Related Work

Recently, several scene classification approaches have been proposed for HRRS images based on the traditional BOW model [18,30]. In a typical pipeline of BOW, we apply the vector quantization method on the extracted local features, to generate a group of clusters using K-means clustering. Each cluster is regarded as a codeword (or visual word) that represents a specific local pattern, and all of the codewords construct a codebook. By mapping the local features to the codebook, we can represent each image scene as an unordered histogram representing the frequency occurrences of codewords. The BOW model is a simple, but effective approach to generate global feature representation for a whole image scene and, thus, remains a very prevalent method for image classification in the computer vision community [31–34]. In order to further improve the discriminative power of BOW, many variants and extensions have been presented. The spatial pyramid matching kernel (SPM) [30] is a classical extension to BOW, which computes a histogram for each subregion of the image and concatenates all of the histograms in a weighted spatial pyramid way. The spatial co-occurrence kernel (SCK) [32], another important extension to the BOW model, considers the spatial distribution of visual words. As an improved version of SCK, the spatial pyramid co-occurrence kernel (SPCK) [7] captures both the absolute and relative spatial arrangements of visual words and achieves good performance on overhead land use scene dataset. Motivated by SPM and SCK, a pyramid-of-spatial-relations model [9] introduces a novel concept to describe quantized relative relationship of a set of local features and outperforms both BOW and SCK. Bolovinou et al. [35] proposed a bag-of-spatio-visual-words model (BoSVW), which incorporates local context information into the BOW representation and can efficiently tackle the problem of high-dimensional spatial feature clustering by introducing the spherical K-means algorithm. In general, all of these methods strongly rely on the extraction of the hand-crafted low-level features, learning the codebook and coding local features, which are usually highly time consuming. In contrast with these BOW-based methods, the proposed fast binary coding method is a unified feature representation framework integrating local feature extraction and feature coding, and thus, it shows priority in computation speed.

The local feature descriptors designed with binarized tricks, which have advantages in adequate robustness while providing high computational efficiency, are very popular in image recognition and face verification application [21,28,36]. Two of the representative features are the local binary pattern (LBP) [21] and the local phase quantization (LPQ) [28], which were originally designed for texture analysis. These two kinds of local feature descriptors are described by assigning a binary code to a pixel's neighborhood. Kannala et al. [36] improved the LBP and LPQ and proposed the binarized statistical image features (BSIF) which is most related to our work. In [36], the binary codes are generated by binarizing the convolutional response of the image and a set of linear filters. We follow the same method as the BSIF to compute image features, but differ in generating the linear filters. In contrast to the BSIF, where the linear filters are only learned via independent component analysis, the proposed FBC comprehensively extends this work and the promising classification performance demonstrates that many unsupervised learning algorithms can learn "good" filters.

Another topic related to our work is unsupervised feature learning (UFL) [37,38], which is capable of automatically learning discriminative features or structures from a large amount of unlabeled data by reasonable unsupervised learning algorithms. A general pipeline of UFL methods is composed of two stages: learning model parameters (usually a dictionary) by a certain unsupervised learning algorithm and encoding input examples to features. Several researchers have applied the UFL methods to the land use scene classification. In [6], a UFL method in which the sparse coding is used for learning sparse features is successfully applied to aerial scene classification. Zhang et al. [39] presents a saliency-guided UFL framework for scene classification. In [39], neural networks are used to train a set of feature extractors, with some techniques to reduce overfitting in the feature learning stage. Hu et al. [40] propose an improved UFL pipeline where both learning model parameters and encoding features are performed on a low-dimensional manifold. It is worth mentioning that the latter two methods are free of any low-level hand-crafted features. In the FBC pipeline, the linear filters are learned by certain unsupervised algorithms, and global features are automatically generated following the binary coding scheme. On this front, the FBC is very similar to UFL methods, where the learned linear filters are equivalent to the dictionary, and the binary coding scheme can be regarded as a special feature encoding stage.

3. Fast Binary Coding For Scene Classification

In this section, we describe the FBC method for extracting the global feature representation of image scenes in detail and theoretically analyze its computational complexity. We also present how the suitable filters are learned via distinct unsupervised learning approaches.

3.1. Fast Binary Coding

Let $I : \Omega \mapsto \mathbb{R}^{d\times M\times N}$ be a d-channel image on the grid $\Omega = \{0, 1, \ldots, M-1\} \times \{0, 1, \ldots, N-1\}$ of size $M \times N$. As we concentrate on describing the scene structures of the image I, we only consider the intensity channel (i.e., the grayscale) in our case, meaning $d = 1$. Let $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$ be K linear filters, with $\mathbf{W}^{(k)} : \Omega_W \mapsto \mathbb{R}^{(2\tau+1)\times(2\tau+1)}$, where $2\tau + 1$ is the number of pixels in one column or row of the filter $\mathbf{W}^{(k)}$. In this subsection, we concentrate on presenting the FBC pipeline,and left the learning of the K linear filters $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$ to Section 3.4. Thus, the k-th filter response $f^{(k)}$ (also referred to as the feature map) is obtained by convolving the image I with the filter $\mathbf{W}^{(k)}$,

$$f^{(k)} = I * \mathbf{W}^{(k)},\ k = 1, \ldots, K \tag{1}$$

where "$*$" denotes the two-dimensional convolution operator,

$$\left(I * \mathbf{W}^{(k)}\right)(x, y) = \sum_{a=-\tau}^{\tau} \sum_{b=-\tau}^{\tau} W^{(k)}(a, b) f(x - a, y - b),\ \forall (x, y) \in \Omega, \tag{2}$$

where x, y denote the pixel position coordinates of the image I. Note that zero-padding is applied to I before the convolution, in order to make the size of $f^{(k)}$ identical to I. Each value $f^{(k)}(x, y)$ in the filter response can be regarded as a descriptor of the local region $\{x-\tau, \ldots, x+\tau\} \times \{y-\tau, \ldots, y+\tau\}$ centered on the corresponding pixel (x, y) in I. Finally, the image scene I outputs K real-valued filter responses, and in other words, we can get a K-dimensional real-valued feature for each pixel. We binarize all of the K responses $\{f^{(k)}\}_{k=1}^{K}$ and obtain the binarized maps $\{B^{(k)}\}_{k=1}^{K}$:

$$B^{(k)} = \hbar_{\varepsilon}\left(f^{(k)}\right),\ k = 1, \ldots, K \tag{3}$$

where $\hbar_{\varepsilon} : v \in R \mapsto \{0, 1\}$ is a threshold function for binarizing the real-valued filter responses with respect to a predefined threshold value ε, which returns one if $v > \varepsilon$ and zero otherwise. When the threshold ε is equal to zero, the threshold function then turns into the Heaviside step function [41]. We take $\varepsilon = 0$ as the default setting in experiments and will discuss the effect on classification performance when the threshold ε varies in Section 5.2.

For each pixel, the K-dimensional real-valued feature is now transformed into a K-bit binary string. We can consider the binary strings as a binary-valued number and convert it back into a single integer value by the following operation,

$$I^{B}(x, y) = \sum_{k=1}^{K} 2^{k-1} \cdot B^{(k)}(x, y),\ \forall (x, y) \in \Omega, \tag{4}$$

where I^{B} is a new generated "image" after the conversion of binary maps. We can note that the value of a pixel in I^{B} is an integer within the range of $[0, 2^{K}-1]$. In analogy to the conventional BOW model, each integer value is regarded as a codeword, and thereby, the size of the codebook results in 2^{K}. A statistical histogram $Y \in \mathbb{R}^{2^{K}}$ is computed on this codebook, which is the resulting global feature representation for image I,

$$Y(m) = \frac{1}{M \times N} \cdot \sum_{(x,y)\in\Omega} \delta\left(I^{B}(x, y), m\right),\ m = 0, 1, \ldots, 2^{K}-1 \tag{5}$$

where $\delta(\cdot, \cdot)$ is a delta function, as $\delta(a, b) = 1$ if a is identical to b and $\delta(a, b) = 0$ otherwise.

In the FBC method, the length of binary strings, i.e., the number of bits, depends on the number of predefined filters K. In addition, the dimension of the final features for the image scene is also closely related to K and exponentially grows with K. Our empirical experience suggests that we should set a relatively small K to make the length of the global features acceptable, and avoid parameters overfitting of the classifier. The whole stage for generating feature representation via FBC is illustrated in Figure 1.

The proposed FBC is actually akin to the BOW model in nature. In FBC, the local features are extracted densely at each pixel via straightforward convolutions; the binary coding stage can be viewed as a simplified version of vector quantization, where the codebook is composed of the consecutive integers, and the local features are then coarsely encoded into these integers (converting the feature vectors to discrete scalar values). Figure 2 shows some examples of binary maps and integer-valued maps. It can be noted that the integer maps still retain abundant useful information that is helpful for recognition, in spite of such a "crude" coding scheme.

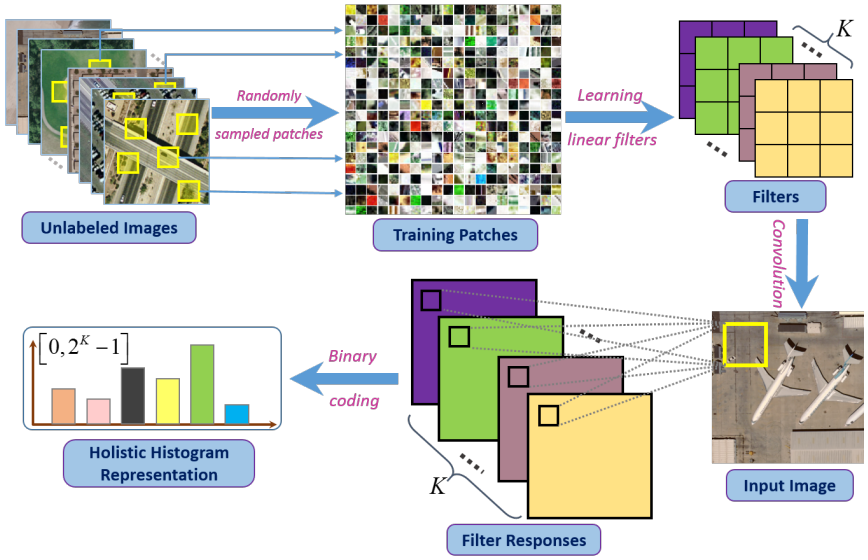

Figure 1. Pipeline illustration of the fast binary coding (FBC) for global feature representation of image scenes. The filters used in the FBC are preliminarily learned from a large amount of randomly-sampled image patches via a specific unsupervised method. We convolve each image scene with the learned filters to generate feature maps and then convert the binary-valued maps to an integer-valued map. Finally, we generate a histogram of features by counting the frequency of each integer.

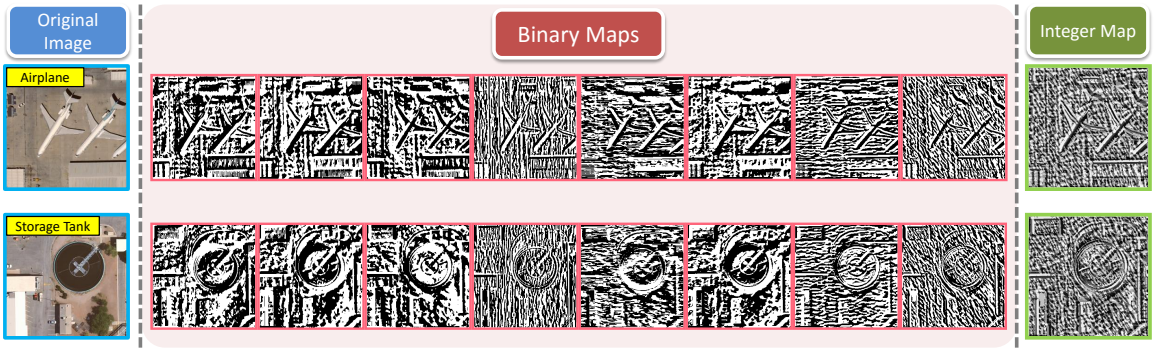

Figure 2. Illustrations of binary-valued maps and integer-valued maps. Here, we use eight filters (of a size of 9×9 pixels) learned by spare coding to generate these binary maps for filter responses. The corresponding integer-valued map is generated by converting each eight-bit binary string into an integer value. The integer maps contain much useful details included in the original images, such as clear edges and corners. These retained details are very informative and discriminative for classification.

3.2. *Analysis of the Computational Complexity of FBC*

It is worth noticing that the overall feature extraction pipeline of the FBC algorithm only contains quite simple mathematical operations at each procedure and can be numerically implemented with high efficiency, as:

- Convolving an image scene with filters is a linear operation;
- Binarizing the filter responses is a thresholding operation;
- Converting binary maps to the integer map is a linear operation according to Equation (4);
- Obtaining the histogram features only needs to count the frequency of integers within $[0, 2^K - 1]$.

All of these light procedures make the FBC method fast for feature representation of scenes. Compared to the FBC algorithm, global feature representation obtained by the BOW model needs to go through complex local feature extraction, time-consuming codebook learning and high nonlinear feature coding. Therefore, it is obvious that the FBC has great superiority over the BOW model in computational complexity intuitively.

To make the comparison of computational cost more clear, as shown in Table 1, we list the concrete computational complexity for each stage of the BOW model and FBC. For the BOW model, the computation cost of the feature extraction stage depends on which kind of local features are extracted, and K-means clustering is used to generate the codebook by default. In the general settings of the BOW model, N_0 and N_1 should be assigned a relatively large value (e.g., $N_0 = 1000$, $N_1 = 50{,}000$), which ensures that a set of more representative visual words is learned. Hence, for the BOW model, not only the local feature extraction is a complicated stage, but the codebook learning and feature coding stage are really slow, as well. In contrast to BOW, the FBC seems to be far more "lighter", because the computational complexity simply lies on the size of image scenes and the number of filters. In fact, the number of filters K is usually set to be a very small value, say $K \leq 12$, for achieving good classification performance. One fundamental reason for the superiority in computational complexity is that the FBC is a unified end-to-end feature extraction method, which can directly generate global feature representation from the original image scene. We present the evaluation of the computation time to verify the low computational cost of the FBC, the details of which are shown in Section 5.2.

Table 1. Comparison of the computational complexity of the FBC and BOW model.

Method	Feature Extraction	Generating Codebook	Histogramming
BOW	-	$T \cdot O(\varrho N_0 N_1)$	$O(\varrho N_0 N_2)$
FBC		$O(MNK)$	

Here, ϱ denotes the length of local features; T denotes the number of iterations of K-means iterations (usually T > 50); N_0 denotes the size of the codebook, i.e., the number of clusters learned by K-means clustering; N_1 denotes the number of local features for learning the codebook; N_2 denotes the number of extracted local features of each image scene; M and N are the size of image scene; K denotes the number of filters.

3.3. Scene Classification

We now can efficiently compute the global features for each image scene in a dataset via the proposed FBC method. These histogram features Y can be directly fed into an off-the-shelf classifier for the classification task. The detailed FBC-based scene classification pipeline is completely presented in Algorithm 1. In this paper, the SVM classifier with the histogram intersection kernel (HIK) is utilized to train and predict labels for new image scenes. Given the histogram feature representation for images i and j as Y_i and Y_j, respectively, the HIK is defined as:

$$K_{HIK}\left(Y_i, Y_j\right) = \sum_{p=1}^{2^K} \min\left(Y_i(p), Y_j(p)\right) \tag{6}$$

Algorithm 1 FBC-based scene classification framework.

Input:

The training set of image scenes and the corresponding ground truth labels, $\mathcal{I}^{tr}$, $\mathcal{L}^{tr}$;
The testing set of image scenes, $\mathcal{I}^{te}$;
A set of learned linear filters, $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$;

Output:

The predicted labels for testing image scenes, $\mathcal{L}^{te}$;

1: For each image scene $I_i^{tr} \in \mathcal{I}^{tr}$, convert it into a gray-level image;
2: Compute filter responses by convolving the image scene with filters $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$;
3: Binarize all of the K response to obtain the binarized maps $\{B^{(k)}\}_{k=1}^{K}$;
4: Convert the binarized maps $\{B^{(k)}\}_{k=1}^{K}$ to an integer-valued map I_i^B, according to Equation (4);
5: Compute a statistical histogram Y_i^{tr} of I_i^B against the codebook of $[0, 2^K - 1]$ integers;
6: Train model parameters of SVM classifier using the holistic histogram representations Y^{tr};
7: For each image in the testing scene $I_i^{te} \in \mathcal{I}^{te}$, compute the histogram Y_i^{te} by repeating Steps 1,2,3,4,5;
8: Predict class labels for all testing image scenes by the trained SVM classifier with Y_i^{te};
9: **return** $\mathcal{L}^{te}$;

3.4. Learning $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$ via Unsupervised Learning Methods

Above, we assume that $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$ is known. This section discusses how to learn such linear filters $\{\mathbf{W}^{(k)}\}_{k=1}^{K}$ for FBC. In the FBC pipeline, it seems that any kind of linear filters can be used to convolve with the image theoretically, and hence, we want to discover what kind of filters are most suitable to FBC feature representation. Inspired by the unsupervised feature learning (UFL) techniques, we attempt to adaptively learn suitable filters $\mathbf{W}^{(k)} : \Omega_W \mapsto R^{(2\tau+1)\times(2\tau+1)}$ by some commonly-used unsupervised learning algorithms. In analogy to the UFL pipeline, we first perform two indispensable pre-processing stages:

- Randomly extract a large number of S image patches with size of $(2\tau + 1) \times (2\tau + 1)$ from the training image dataset;
- Normalize each patch to zero mean and unit variance.

After pre-processing all sampled image patches, we can train a dictionary/basis D via a proper unsupervised learning method. In fact, the learned dictionary/basis is another pattern of linear filters to some extent; hence, we can readily obtain the set of filters needed in the FBC, by choosing the right number of entities from the dictionary/basis and resizing the entities to the size of image patch samples. Here, we present how we learn suitable linear filters by unsupervised learning algorithms. Eight conventional algorithms that will be tested in subsequent experiments are briefly introduced below.

3.4.1. K-Means Clustering

Given the normalized patch vectors $x^{(i)}, i = 1, 2, \cdots, S$, K-means can learn a dictionary D containing K cluster centers. The objective function is formulated by minimizing the distance between training samples and their cluster centroids, which is defined as:

$$\begin{aligned} &\min_{D,c} \sum_{i}^{S} \|Dc^{(i)} - x^{(i)}\|_2^2 \\ &s.t. \, \|D^{(k)}\|_2 = 1, \, \forall k, \text{and}, \, \|c^{(i)}\|_0 = 1, \, \forall i \end{aligned} \tag{7}$$

where $c^{(i)}$ is the assignment vector (or code vector) of the sample $x^{(i)}$ to the clusters and $\|c^{(i)}\|_0$ is the number of non-zero elements in $c^{(i)}$. This objective is optimized by an alternating iteration over $c^{(i)}$

and D. Specifically, each learned centroid $D^{(k)}$ can be regarded as a filter $\mathbf{W}^{(k)}$ by simply resizing it to the original size of the image patch.

3.4.2. Orthogonal Matching Pursuit

We generalize the optimization problem for K-means described above by allowing more than one non-zero element in $c^{(i)}$. This can enable each code vector to represent more complex patterns. The improved optimization objective is defined by modifying Equation (8):

$$\begin{aligned} &\min_{D,c} \sum_{i}^{S} \|Dc^{(i)} - x^{(i)}\|_2^2 \\ &s.t. \, \|D^{(k)}\|_2 = 1, \, \forall k, \text{and}, \, \|c^{(i)}\|_0 \leq \Lambda, \, \forall i \end{aligned} \tag{8}$$

where Λ is the maximal number of non-zero elements allowed to recover each $x^{(i)}$. However, this problem is difficult to solve because of the non-convex property of the constraints. In order to perform an alternating optimization like K-means, we utilize orthogonal matching pursuit (OMP) [42] to compute code vector $c^{(i)}$ with at most Λ non-zeros (also known as OMP-Λ). Each entry $D^{(k)}$ of the learned basis D can be naturally considered as a linear filter $\mathbf{W}^{(k)}$ like the one used in the K-means case.

3.4.3. Sparse Coding

Sparse coding (SC) [43] is a biologically-inspired algorithm that aims to find a set of basis vectors $D(k)$, such that an input sample $x(i)$ can be represented as a linear combination of these basis vectors. All of the basis vectors form a dictionary. Given the normalized patch vectors $x^{(i)}, i = 1, 2, \cdots, S$, the objective of learning the dictionary D in the sparse coding scheme can be defined as:

$$\begin{aligned} &\min_{D,\alpha} \sum_{i}^{S} \|D\alpha^{(i)} - x^{(i)}\|_2^2 + \lambda \|\alpha^{(i)}\|_1 \\ &s.t. \, \|D^{(k)}\|_2 \leq 1, \, \forall k \end{aligned} \tag{9}$$

where α denotes the sparse vectors and λ is the penalty weight. The L^1-norm penalty encourages more zero elements in $\alpha^{(i)}$ controlled by λ. We can easily optimize this objective by online learning techniques [44] in recent years. When accomplishing the learning stage, each basis vector $D^{(k)}$ in the dictionary D becomes a learned linear filter $\mathbf{W}^{(k)}$.

3.4.4. Non-Negative Matrix Factorization

Non-negative matrix factorization (NMF) was originally proposed for parts-based decomposition of images and can be interpreted as a relaxed form of K-means. Very similar to the optimization formulation, NMF minimizes the following cost:

$$\begin{aligned} &\min_{D,\alpha} \sum_{i}^{S} \|x^{(i)} - D\alpha^{(i)}\|_2^2 \\ &s.t. \, D^{(k)} \geq 0, \, \forall k, \text{and}, \, \alpha^{(i)} \geq 0, \, \forall i \end{aligned} \tag{10}$$

where matrix D and the vectors $\alpha^{(i)}$ are forced to have non-negative components. The methods for addressing this problem are seen [45]. In practice, each entry of the factor matrix D is viewed as a linear filter $\mathbf{W}^{(k)}$.

3.4.5. Gaussian Mixture Model

Another unsupervised clustering algorithm is the Gaussian mixture model (GMM), which supposes that the data are generated from K Gaussian distributions:

$$P(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}_k(x; \mu_k, \Sigma_k) \tag{11}$$

where π_k are prior probabilities of x sampled from each distribution and μ_k, Σ_k are the mean and covariance of the k-th component distribution. Expectation-maximization (EM) can efficiently estimate parameters (μ, Σ, π). In the FBC case, we view the mean μ of each distribution as the linear filter $\mathbf{W}^{(k)}$.

3.4.6. Principal Component Analysis

PCA is probably the most common unsupervised linear dimensionality reduction technique. The main goal of PCA is to iteratively find orthogonal directions maximizing the variance of samples or it can be cast as a low-rank matrix factorization problem:

$$\begin{aligned} &\min_{U} \|X - UU^\top X\|_F^2 \\ &s.t.\, UU^\top = \mathbf{I} \end{aligned} \tag{12}$$

where U is an orthonormal matrix, $\mathbf{I}$ is the identity matrix and X is a matrix of concatenating all training patch vectors, i.e., $X = [x^{(1)}, x^{(2)}, \ldots, x^{(S)}]$. The solution to Equation (12) is that the columns of matrix U are the eigenvectors of matrix $XX^\top$. We view the first K principal eigenvectors of matrix $XX^\top$ (i.e., the first K columns of U) as the set of linear filters.

3.4.7. Locality-Preserving Projection

Locality-preserving projection (LPP) [46] is another widely-used linear dimensionality-reduction algorithm and is also a linear approximation to the classic Laplacian eigenmap (LE). LPP aims to find a linear mapping matrix $\mathcal{M}$ that can embed input data on a low dimensional space. $\mathcal{M}$ can be generated by solving the following generalized eigenvalue problem:

$$XLX^\top \mathbf{m} = \lambda XDX^\top \mathbf{m} \tag{13}$$

where X is the set of input patch vectors, λ, L and D denote the eigenvalue, Laplacian matrix and diagonal matrix, defined identically in LE, and $\mathbf{m}$ is the eigenvector. We choose the first K eigenvectors as linear filters according to the eigenvalues in ascending order.

3.4.8. Auto-Encoder

An auto-encoder (AE) [37,47] is a special structure of neural network consisting of input layer x, hidden layer z and output layer $\hat{x}$. We compute hidden layer $z = S(D^{(1)}x + b^{(1)})$ with trainable parameters $D^{(1)}$ and $b^{(1)}$, where $S(\cdot)$ is a nonlinear function. The output layer is then computed by a similar affine transformation $\hat{x} = D^{(2)}z + b^{(2)}$. The objective is formulated by ensuring $\hat{x}$ to be a good reconstruction of input x:

$$\min_{D^{(\cdot)}, b^{(\cdot)}} \sum_{i}^{S} \|D^{(2)} S(D^{(1)} x^{(i)} + b^{(1)}) + b^{(2)} - \hat{x}^{(i)}\|_2^2 \tag{14}$$

We can use the off-the-shelf numerical solver to train this single layer network with the gradients computed by the back propagation algorithm. Each column of the weight matrix $D^{(1)}$ is used as a learned linear filter $\mathbf{W}^{(k)}$.

The approach of learning filters via independent component analysis (ICA) [48] has been provided in [36], so we do not repeat the details here. The visualization results of the various learned filters are shown in Figure 3. It is obvious that each set of filters shows distinctive responsive characteristics to others.

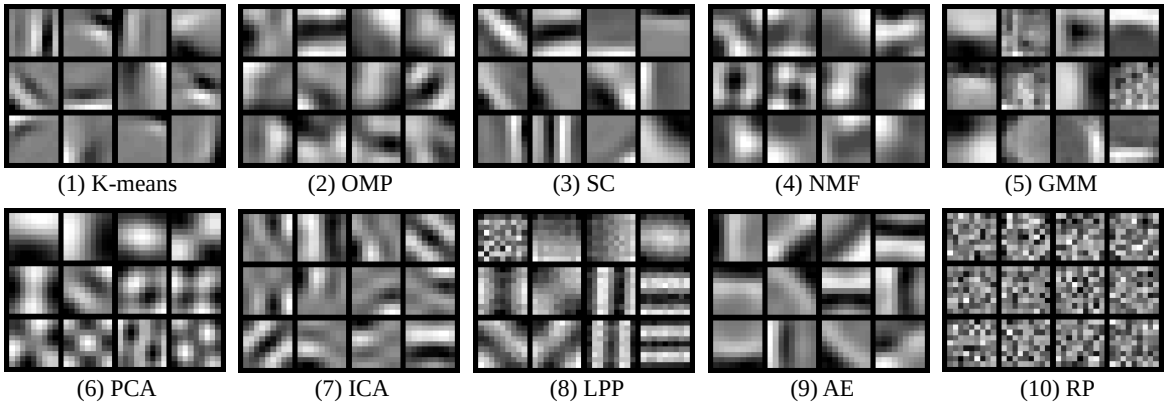

Figure 3. Examples of a set of learned filters using different unsupervised leaning methods from large quantities of image patches. Each entity of the learned basis is resized to the size of image patches (10×10 pixels). Note that the some Gabor-like filters can be achieved by K-means, orthogonal matching pursuit (OMP), sparse coding (SC) and auto-encoder (AE). The noise-like filters shown in (10), called random projection (RP) filters [49], are generated from an independent zero-mean, unit-variance normal distribution.

4. Extensions to FBC

Although the proposed FBC can generate holistic feature representations for image scenes in an efficient way, there still remain two evident defects in the FBC pipeline:

1. FBC lacks sufficient representative power to depict the spatial layout information of image scenes;
2. FBC counts all of the codewords to construct the histogram feature, whereas some of the codewords are probably not helpful to the descriptive ability of histogram features and even have a negative influence on the final classification performance.

Therefore, in terms of these two aspects, we try to improve the FBC with elaborate methods that are able to consider spatial context information or to provide good codeword candidates. In this section, the spatial co-occurrence kernel and saliency-map-based coding scheme, as two extensions to the FBC, are discussed.

4.1. Spatial Co-Occurrence Kernel

As described previously, we can generate a new "coding image" I^B by converting the binary map to integer values. Now, each pixel in the I^B can be viewed as a codeword $p_i \in D$ at pixel location (x_i, y_i), and obviously, the size of D is 2^K according the coding scheme of FBC. Following the instruction of [32], which discovers the spatial dependence of the visual words inspired by gray level co-occurrence matrices (GLCM), a codeword co-occurrence matrix (WCM) for I^B is defined as:

$$WCM(m,n) = \#\{(p_i, p_j) | (p_i = m) \wedge (p_j = n) \wedge dist(p_i, p_j)\} \tag{15}$$

$$dist(p_i, p_j) = \begin{cases} 1 & \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \leq r \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

where $\#\{\cdot\}$ denotes the cardinality of set and r is a constant measuring the co-occurrence property of two codewords. Figure 4 illustrates the scheme of computing the WCM. The definition shows that WCM computes the number of codeword pairs that satisfy the spatial distance constraint, and the size of WCM leads to $2^K \times 2^K$. Such a definition also infers that WCM is a symmetric matrix. The visualized examples of WCMs are displayed in Figure 5.

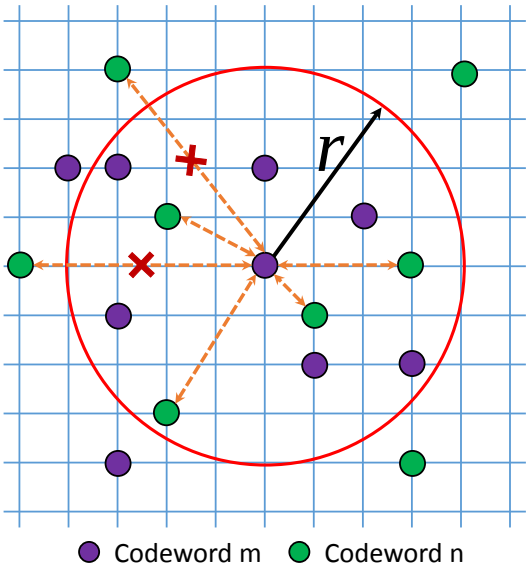

Figure 4. Illustration of computing the spatial codeword co-occurrence matrix $WCM(m,n)$. Only the codeword pairs within the distance constraint r are valid. In this way, we can discover the local spatial properties of the coding image.

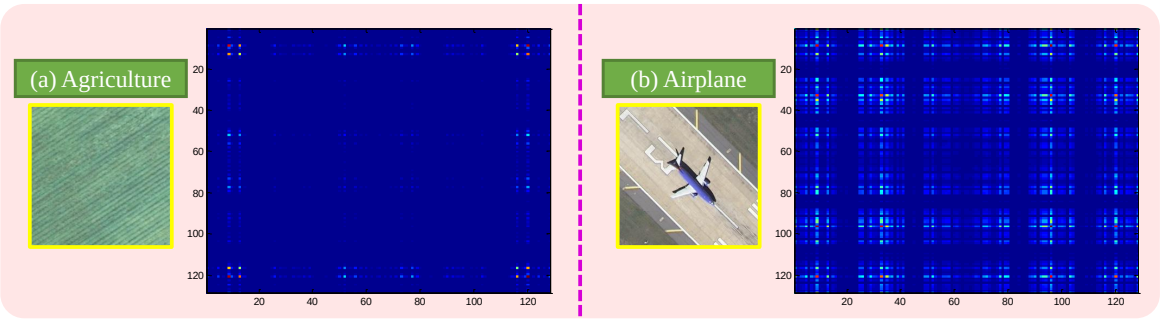

Figure 5. Two visualized examples of WCMs. The WCMs for airplane and agriculture show a distinct appearance, which means that the WCM itself has some discriminative power for representing the image scene. The highlights in WCM denote the frequent pairwise patterns of codewords. (**a**) Agriculture; (**b**) Airplane.

The spatial co-occurrence kernel is computed from the intersection of two WCMs, analogous to the definition of HIK:

$$K_{SCK}(WCM_i, WCM_j) = \sum_m \sum_n \min\big(WCM_i(m,n), WCM_j(m,n)\big), \tag{17}$$

where WCM_i, WCM_j is the final WCM for image i and j. Although the SCK can be solely used for SVM instead of HIK, we still can compute a joint kernel combining the SCK and HIK together:

$$K_{Joint}\big(\{Y_i, WCM_i\}, \{Y_j, WCM_j\}\big) = K_{HIK}(Y_i, Y_j) + K_{SCK}(WCM_i, WCM_j). \tag{18}$$

The codebook used for the WCM can be different from the codebook for generating histogram representations. Otherwise, we perform $L2$ normalization to the histogram Y and WCM before computing these kernels, since the scale of Y and WCM between images is not uniform.

4.2. Feature Coding Based on the Saliency Map

In the FBC pipeline,the feature maps of each image scene are generated by convolving the image with a set of filters. Actually, the "convolution operation" of the image indicates that we extract local features via a dense sampling strategy with the sampling stride being one pixel.

Recently, saliency detection has been widely used in many computer vision applications, and in particular, several works [39,50,51] have reported that saliency-based sampling can help to improve the performance in scene classification. Visual saliency is the perceptual ability of a vision system (e.g., human) to focus on the interesting information while ignoring irrelevant information. Saliency detection is to detect the most attention-grabbing objects or salient regions that stand out relative to their neighbors in a scene. Generally, saliency detection models output a saliency map where each pixel value measures the degree of saliency belong to the salient object. Zhang et al. [39] were first to introduce a context-aware [52] saliency detection algorithm to guide the sampling in scene classification of high-resolution remote sensing images and show its effectiveness in a UFL-based classification framework. However, it remains unclear how this strategy can act on the performance in other classification framework. In contrast with [39], where only one saliency detection method (context-aware) is evaluated, we investigate fifteen kinds of typical methods, which are AC [53], *attention by information maximization* (AIM) [54], *context-aware* (CA) [52], *context-based* (CB) [55], *discriminative regional feature integration* (DRFI) [56], *frequency-tuned* (FT) [57], *graph-based visual saliency* (GBVS) [58], IM [59], *low rank matrix recovery* (LRR) [60], *maximum symmetric surround* (MSS) [61], *rarity-based saliency* (RARE) [62], *salient segmentation* (SEG) [63], *self-resemblance* (SeR) [64], *spectral residual* (SR) [65] and *saliency using natural statistics* (SUN) [66], and present detailed comparative experiments for them. We do not review these methods due to the limited space, and details of each method can be found in its corresponding paper. The saliency map examples generated by these state-of-the-art methods are shown in Figure 6.

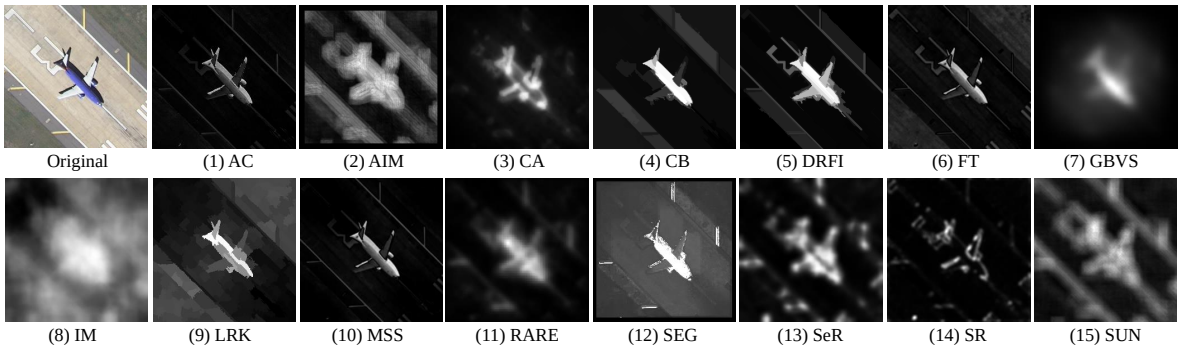

Figure 6. Saliency maps computed by different state-of-the-art saliency detection methods (1–15). Most maps highlight the edges of the salient object (airplane), and are of low resolution compared to the original scene.

Note that the saliency map I^{sal} of each image scene has the same size as its coding image I^B, and thus, we try to use to the saliency map of a scene to guide the coding strategy of histogram representations: only the codewords whose saliency value exceeds a threshold σ are counted in the histogram, while the rest of the codewords are left out. We define the threshold σ as:

$$\sigma = mean\left(I^{sal}\right) - \lambda\left(mean\left(I^{sal}\right) - min\left(I^{sal}\right)\right) \tag{19}$$

where $\lambda \in [0,1]$ is a scale parameter and $mean(\cdot)$ and $min(\cdot)$ denote the mean and minimum of I^{sal}. When $\lambda = 0$, we obtain the maximal threshold. As the λ increases, more codewords contribute to build the histogram representation. The saliency map has no effect on the coding map as $\lambda = 1$.

5. Experiments and Analysis

In this section, we provide the detailed experimental setup, and the results on two public datasets are also discussed. Additionally, numerous experiments are presented to show how the SCK and the saliency maps exactly influence classification performance.

5.1. Experimental Setup

We evaluate the proposed FBC method on two public land use scene datasets, which are:

- *UC Merced Land Use Dataset*. The UC Merced dataset (UCM) [32] contains 21 typical scene categories, each of which consists of 100 images with a size of 256 × 256 pixels. Figure 7 shows two examples of each class included in this dataset. This dataset shows very small inter-class diversity among some categories that share a few similar objects or textural patterns (e.g., dense residential and medium residential), which leads the UCM dataset to be a challenging one.
- *WHU-RS Dataset*. The WHU-RSdataset [4] is a new publicly-available dataset, which consists of 950 images with a size of 600 × 600 pixels uniformly distributed in 19 scene classes. All images are collected from Google Earth (Google Inc.). Some example images are shown in Figure 8. We can see that the variation of illumination, scale, resolution and viewpoint-dependent appearance makes each scene category more complicated than the UCM dataset.

Figure 7. Two examples of each scene category in the UC Merced dataset.

We introduce ten commonly-used unsupervised learning methods, which are GMM, K-means, PCA, ICA, LPP, SC, OMP, AE, NMF and random projection (RP) [49], to learn a set of filters prepared for FBC, aiming to find the most appropriate method that can explore the natural statistical properties of image patches. These methods can learn different linear filters that have various filtering properties. Since we only consider two-dimensional filters learned from image patches in grayscale, thus, all of the original colored image scenes (with RGB channels) in the dataset are first converted to grayscale. In our experiments, the randomly-sampled image patches used for learning filters are beforehand mean-removed. The threshold ε used for binarizing filter responses is fixed to be 0, if not specified. At the classification stage, all of the classes included in the dataset are used for classification, and we randomly choose samples of each class for training and make the rest for testing: the ratio of the training set/testing set is 4:1 for the UCM dataset and 3:2 for the WHU-RS dataset. An off-the-shelf SVM library [67] is used for SVM training and testing. The resulting average classification accuracy is obtained over 100 runs.

Some open source libraries are used to implement the unsupervised learning methods mentioned above: VLFeat [68] for K-means and GMM; SPAMS [44] for SC, OMP and NMF; the DR toolbox [69] for PCA, LPP and AE; publicly available ICA package [70] for ICA. All experiments in

our work are implemented via MATLAB on the Windows 7 platform, with a 2.93-GHz hexa-core Intel Xeon CPU.

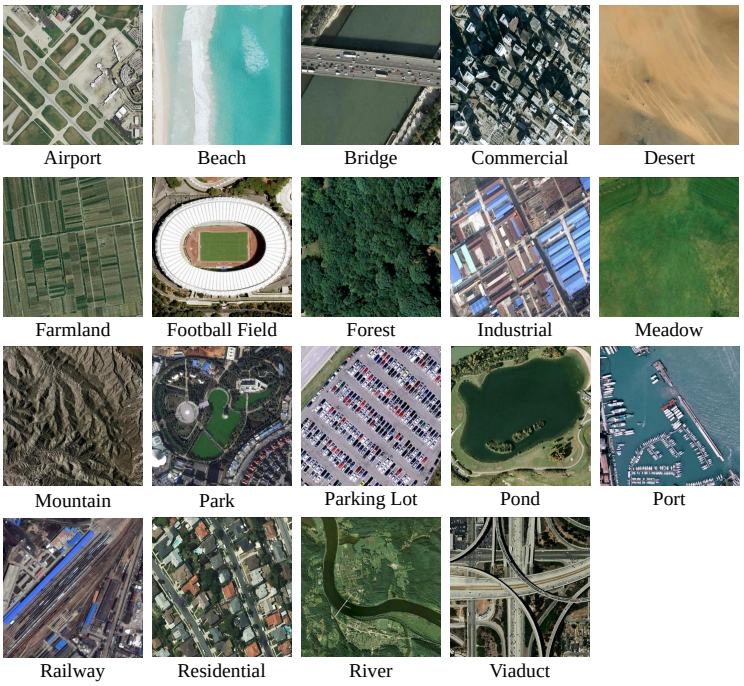

Figure 8. Examples of each category in WHU-RS dataset.

5.2. Experimental Results

5.2.1. Results of FBC

The classification results with different learned filters (the filter size is set to 9×9) are reported in Figure 9. The results show that ICA performs better than other unsupervised techniques consistently on two datasets when the filter size is less than or equal to nine. When the filter size is greater than nine, PCA, NMF, LPP and ICA show comparative performance on the UCM dataset; LPP has the leading performance on the WHU-RS dataset. We can also note that the GMM obviously leads to the worst performance. This is probably because only using mean vectors of each Gaussian component is not an optimal way of generating filters in the GMM case and seriously weakens the advantage of the GMM. On the whole, the performance gradually improves as the number of filter increases. In addition, it is amazing that the RP whose filters are generated from zero-mean, unit-variance normal distribution achieves the best mean accuracies with 83.45% in all experiments. This encouraging result indicates that we can easily obtain a set of suitable filters without any learning procedure. In fact, there are several works [11,49,71,72] that have also shown similar observations that random filters can be very effective in texture and scene classification, and our promising results of RP filters further support this evidence. We also compare the capability of the hand-engineered filters with the learned filters in the FBC framework. The maximum response filters (MR) [73] and the Schmid filters (S) [74], which are specially designed for texture recognition, are investigated; the results in Figure 9a show that a majority of learned filters perform far better (over 10 percent accuracy) than these two specially-designed filters.

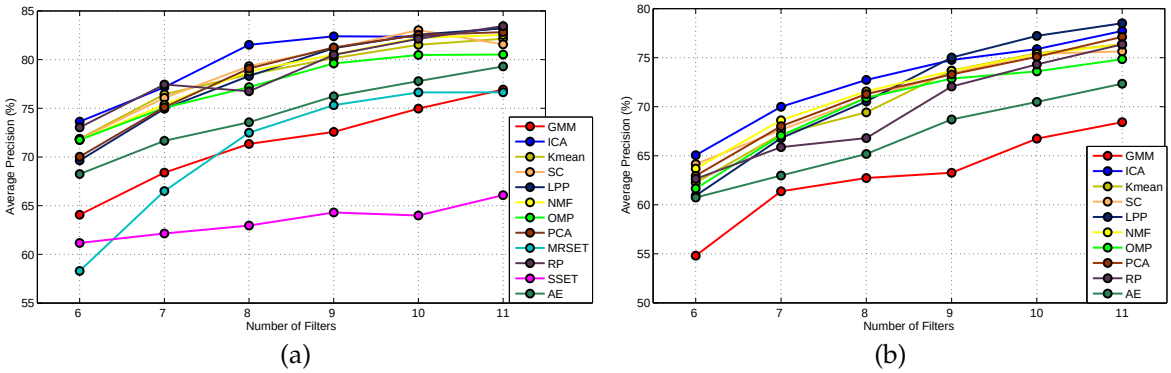

Figure 9. Classification results on two datasets with different learned linear filters. The MRSET and SSET denote the set of maximum response (MR) filters and the Schmid (S) filters, respectively, both of which are hand-designed filters and were originally designed for texture classification task. Here, we not only compare the filters learned by different unsupervised algorithms, but also compare the learned filters with the hand-designed filters in the FBC framework. (**a**) Classification accuracies on the UCM dataset; (**b**) classification accuracies on the WHU-RS dataset.

The effect of the filter size on classification performance is shown in Figure 10. Filters learned by K-means and SC are respectively evaluated here. The results show that classification performance consistently improves as the number of filters grows, except for some rare cases. In general, we can see that performance severely decreases with too small or too large a filter size. The optimal size of filter by K-means and SC is different: filters of a size of 5×5 yield the best accuracy when learned by K-means; filters of a size of 7×7, 9×9, 11×11 lead to comparative performance when learned by SC. We can infer that filters learned via different learning algorithms determine differentially the best filter sizes.

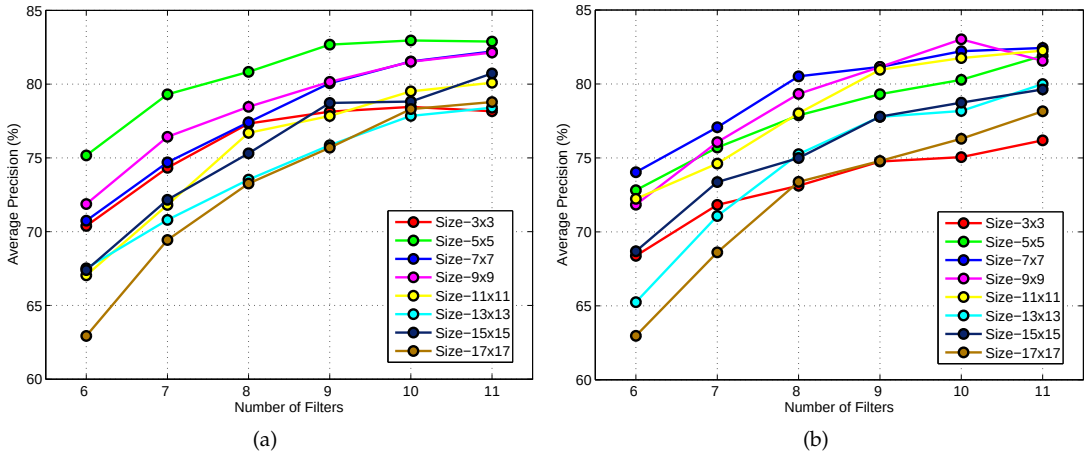

Figure 10. Results with different filter sizes when the filters are learned via K-means and sparse coding. (**a**) Classification results with K-means filters; (**b**) classification results with sparse coding filters.

In the above experiments, the threshold value ε for binarizing filter responses is empirically fixed to be zero. In fact, the threshold ε is a key parameter in FBC, because it directly controls all of the generated binary maps and further relates to the final image representations. Hence, we also investigate the effect of the binarization threshold on classification performance, and the results are shown in Figure 11. An interesting observation from the results is that the final performance changes highly symmetrically when ε varies from -100–100. It appears only the absolute value of ε

influences performance. We can also note that the performance increases first and then gradually drops with the increase of $|\varepsilon|$. The best performance consistently occurs at $|\varepsilon| = 5$ for the two datasets. Thus, in contrast to the previous setting of $\varepsilon = 0$, we can achieve an obvious performance gain (especially for the WHU-RS dataset) at $|\varepsilon| = 5$. This reveals that the threshold ε indeed has a substantial impact on final performance, and the empirical value $\varepsilon = 0$ is not the best choice for the two datasets.

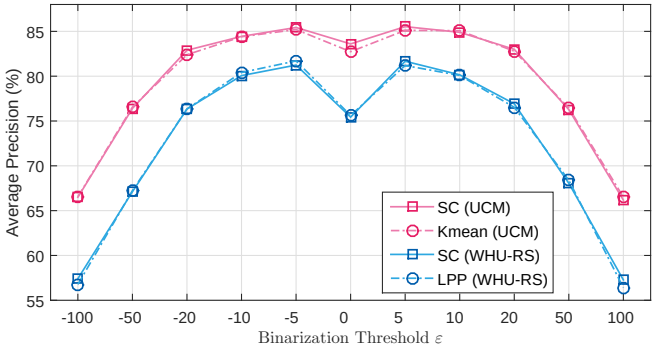

Figure 11. The effect of binarization threshold ε on classification performance. Here, the SC and K-means filters are tested on the UCM dataset; the SC and locality-preserving projection (LPP) filters are tested on the WHU-RS dataset. We evaluate the classification accuracy with 10 filters of a size of 9×9 when the threshold ε varies within a wide range.

Note that when the filters are obtained, the whole stage of feature representation by FBC for images scenes is fairly fast because FBC is a totally feed-forward process and does not contain the extraction of low-level features, as well as complex feature encoding and pooling steps compared to the typical scene classification pipeline. Figure 12 shows the run time of the FBC for generating global features of all image scenes in the UCM data. We achieve the results by testing filters learned by SC with different sizes and numbers. We can note that time consumption increases approximately linearly with the size and number of filters. At the best performance point with 10 filters of 9×9 (according to Figure 10b), it only takes less than 50 s to extract features for all 2100 images of the UCM dataset, and this is beyond $60\times$ faster than the BOW model (taking more than one hour on our computer). This intuitive evaluation on the running time experimentally demonstrates the superiority of the FBC in the aspect of computational complexity. The promising results also show that FBC has fair potential to become a practical industrial scene classification tool.

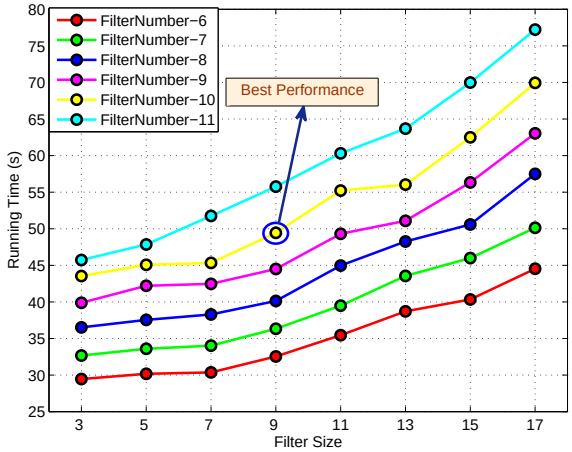

Figure 12. Time consumption of global feature representation with different filter sizes and filter numbers on the UCM dataset. The coordinate point of "best performance" represents that when the SC filters are tested, we achieve the best classification accuracy with 10 filters of a size of 9×9 pixels.

We also compare our method to the off-the-shelf methods that have reported classification accuracy on the UCM dataset, shown in Table 2. We can note that our method can outperform the traditional BOW-based methods (BOW, SPM, SCK, SPCK++), as well as the newly-published methods based on unsupervised feature learning (SC + pooling, SG+ UFL). The encouraging results demonstrate the FBC is apparently superior to the BOW model and the UFL-based model, both in accuracy and computational cost. Although COPD [12], which is an object-oriented scene classification framework, presents better performance than ours, it contains very complex pre-training to discover discriminative visual parts from images in the aspect of computational efficiency. The UFL-SC [40], which outperforms our method, also encompasses time-consuming manifold learning, nonlinear dictionary learning and a feature encoding stage. In other words, the proposed FBC is a compromise method that achieves a balance in accuracy and efficiency. Moreover, due to the surprising speed of representing image scenes, the FBC can be hopefully applied to a practical scene analyzing system for quickly predicting scene classes.

Table 2. Mean classification accuracy comparison on the UCM dataset. SPM, spatial pyramid matching kernel; SCK, spatial co-occurrence kernel; SPCK, spatial pyramid co-occurrence kernel; UFL, unsupervised feature learning.

Methods	Classification (%)
BOW [7]	71.86
SPM [30]	74
SCK [32]	72.52
SPCK++ [7]	77.38
SC + pooling [6]	81.67 ± 1.23
SG+ UFL [39]	82.72 ± 1.18
NFNC [71]	87.67
UFL-SC [40]	90.26 ± 1.51
COPD [12]	91.33 ± 1.11
FBC ($\varepsilon = 0$)	$\mathbf{83.45 \pm 1.6}$
FBC ($\lvert\varepsilon\rvert = 5$)	$\mathbf{85.53 \pm 1.24}$

5.2.2. Results of SCK

We have evaluated the performance of HIK, SCK and the joint kernel on the basis of the proposed FBC. Since the size of the codebook is determined by the number of filters K, i.e., 2^K, the K also plays an important role in the performance of SCK and the joint kernel. For each dataset, two codebooks of different sizes are used to compute the co-occurrence matrices for SCK: 64 and 128 for the UCM dataset (six and seven filters are used accordingly); 128 and 256 for the WHU-RS dataset (seven and eight filters are used). Here, we only study the cases of filters learned by K-means and SC. The constant parameter r is set to 50 empirically through all experiments.

Table 3 presents the accuracies of only SCK used for classification on the two datasets, and we note that SCK performs better than HIK on UCM and worse than HIK on WHU-RS for both dictionary sizes. The joint kernel, i.e., the SCK combined with HIK, outperforms the single HIK consistently on the two datasets, shown in Figure 13. It is obvious that the joint kernel improves performance notably when the number of filters K is small, but the performance gap is gradually reduced as K increases. We can see that when $K = 10, 11$, the performances of the two kernels are very close on the UCM dataset. Figure 13 also shows us that the joint kernels with different SCK have very similar performance along with K varying, which is especially reflected in WHU-RS. On the whole, the SCK is a helpful spatial extension to the traditional non-spatial HIK for scene classification.

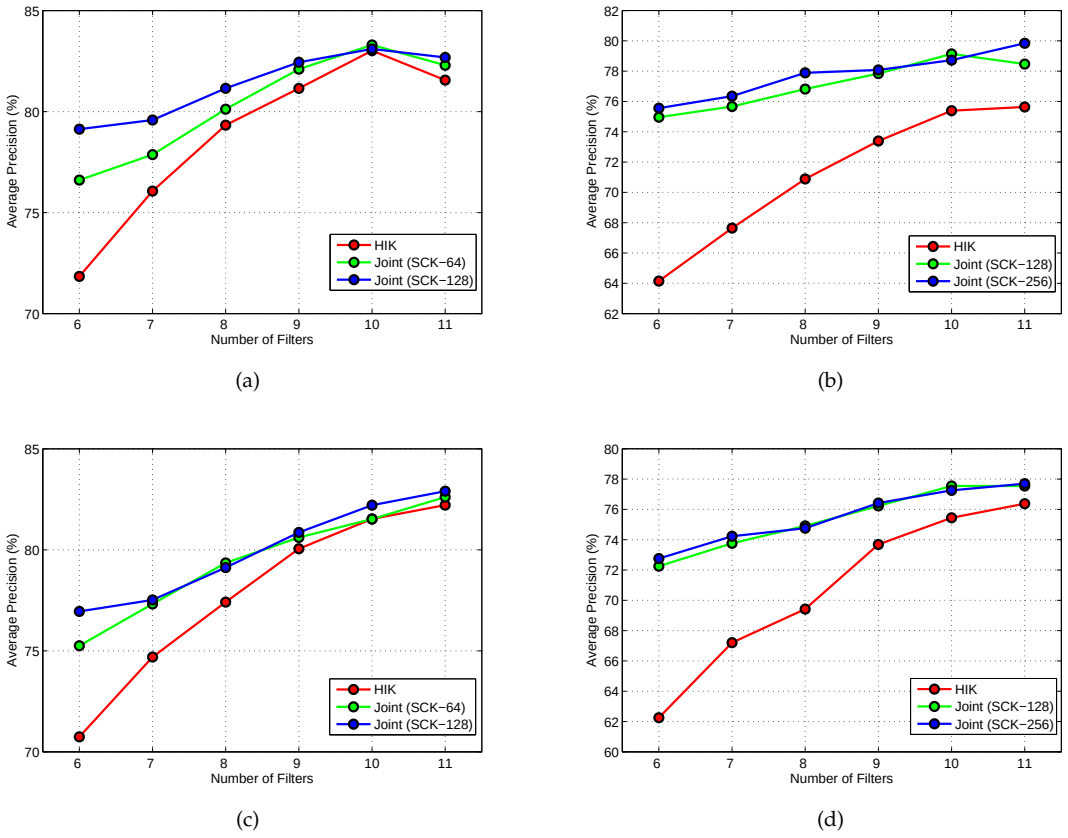

Figure 13. Results on two datasets with the histogram intersection kernel (HIK) and joint kernels with different filter settings. (**a**) Results on the UCM dataset with SC filters (9×9 pixels); (**b**) results on the WHU-RS dataset with SC filters (9×9 pixels); (**c**) results on the UCM dataset with K-means filters (7×7 pixels); (**d**) results on the WHU-RS dataset with K-means filters (7×7 pixels).

Table 3. Performance comparison between SCK and HIK.

	UCM Dataset				**WHU-RS Dataset**			
Codebook Size	64		128		128		256	
Kernel Type	SCK	HIK	SCK	HIK	SCK	HIK	SCK	HIK
K-Means Filters	75.73 ± 1.85	71.87 ± 2.00	78.05 ± 1.96	74.69 ± 1.90	58.67 ± 2.07	67.21 ± 1.99	59.54 ± 2.19	69.42 ± 2.04
SC Filters	77.21 ± 1.94	71.84 ± 1.88	80.31 ± 1.69	76.06 ± 1.89	60.64 ± 2.25	67.64 ± 1.78	63.26 ± 1.91	70.88 ± 2.14

5.2.3. Results of Saliency-Based Coding

In this part, we evaluate fifteen kinds of saliency detection algorithms and discover how their saliency maps affect the classification performance under different saliency thresholds. We conduct experiments only on the UCM dataset (certainly, these trials can be replicated on WHU-RS or other datasets) and fix the training and testing set for the sake of concentrating only on the effect of the saliency map on performance. Ten filters of a size of 9×9 learned by sparse coding are used to generate the coding images for all image scenes. In this parameter configuration, the classification accuracy through the standard FBC pipeline is 84.29%, which is considered as a baseline accuracy.

The performances of the 15 saliency detection methods are shown in Table 4, where 10 saliency threshold scales are tested. We can see that the GBVS leads to the best performance overall among all methods. For each method, although fluctuating in some cases, the performance gradually improves as λ increases in general. This is because more non-salient codewords that may be helpful to the representative power of the histogram are counted when the saliency threshold becomes small, especially for some scene categories, which consist of textural structures and do not contain any

salient objects or region, such as agricultural and chaparral. The results also show that we should set a proper λ in order to get a better performance than the baseline. The best λ is varying for each saliency detection model. It is worth noticing that some models lead to worse performance than the baseline whatever the λ is set to, e.g., AC, CB and SR. This reveals that not all of the saliency detection methods can work as a beneficial guide to the coding stage in the FBC-based classification scenario.

Table 4. Comparison of the classification performance of saliency-based coding via different saliency detection methods. AIM, attention by information maximization; CA, context-aware; CB, context-based; DRFI, discriminative regional feature integration; FT, frequency-tuned; GBVS, graph-based visual saliency; LRR, low rank matrix recovery; MSS, maximum symmetric surround; RARE, rarity-based saliency; SEG, salient segmentation; SeR, self-resemblance; SR, spectral residual; SUN, saliency using natural statistics.

	Saliency Detection Methods														
scale λ	**AC**	**AIM**	**CA**	**CB**	**DRFI**	**FT**	**GBVS**	**IM**	**LRR**	**MSS**	**RARE**	**SEG**	**SeR**	**SR**	**SUN**
0	79.29	83.1	73.33	71.19	77.38	78.57	84.05	77.62	75	78.81	74.52	81.9	78.81	74.76	75.71
0.1	80.71	83.1	76.9	73.33	78.81	79.05	83.33	79.05	75.71	80.71	74.29	81.67	80.71	75.24	76.9
0.2	81.19	82.86	77.14	72.38	77.86	79.05	84.52	79.76	78.33	81.19	75.24	84.76	80.48	76.19	78.1
0.3	81.9	82.14	79.76	72.14	77.38	79.52	85.24	81.43	78.81	80.71	77.62	85.48	80.24	77.62	80
0.4	81.67	83.1	80	72.38	79.29	80.71	84.76	82.14	81.9	83.1	76.9	84.52	81.9	78.57	81.19
0.5	81.9	83.81	83.1	72.38	80	81.43	83.81	83.1	83.1	83.1	80	84.29	82.62	79.76	83.81
0.6	81.43	84.52	83.1	74.05	79.05	82.62	83.81	83.33	83.57	84.52	81.43	84.29	82.14	81.67	83.57
0.7	82.38	84.76	83.57	74.76	78.1	82.86	84.76	84.76	85.48	84.29	82.86	84.52	84.29	83.1	84.29
0.8	82.38	84.76	84.76	75.24	79.05	84.05	85.24	85.48	84.05	84.52	83.81	84.29	84.29	82.86	83.81
0.9	81.67	83.57	85	79.76	82.62	83.33	85.48	84.76	85	83.57	84.29	84.29	84.29	84.05	84.29

6. Discussion

From the comparative experiments of different learned filters in Figure 9, we can observe that the learned filters outperform the hand-designed filters by a large margin, showing that we can obtain statistically meaningful structures or patterns from local image patches by employing unsupervised learning methods rather than manual design. Except for the GMM and AE, other methods evaluated in this paper generally result in comparable performance. A fairly interesting observation is that even with random filters, the FBC can still achieve impressive performance. This characteristic makes FBC a versatile method that can be straightforwardly adapted for new datasets because the FBC does not need any learning strategies when generating image representations.

We can also note from Figure 10 that the two parameters, i.e., the size of filters and the number of filters, have a large impact on final performance. It shows a stable upward trend in performance with the increasing number of filters. However, too large a number of filters is not desirable: on the one hand, it increases the computational cost with limited performance gain; on the other hand, it exponentially increases the dimensionality of the final image representations (as described previously, the dimension of the image representation is 2^K) and, thus, results in highly redundant features that may be adverse at the stage of the training classifier. Our experiments show that the number of filters ranging from 9–11 is suitable for the tradeoff of speed/accuracy. In addition, it appears that we suffer a great loss with too small or too large a size of filters. The filter size ranging from 5×5–11×11 can always result in good performance.

Figure 13 and Table 3 show that although the SCK does not always outperform HIK, combining SCK and HIK consistently leads to performance gain and demonstrates that the SCK, which considers relative spatial information indeed, helps to improve classification performance in the pipeline of FBC. Table 4 shows that not all of the saliency detection methods can work as a beneficial guide to the coding stage in the FBC pipeline. With proper parameter settings, we can obtain better performance than the original FBC. For instance, when $\lambda = 0.9$, the GBVS results in an accuracy of 85.48%. Another interesting observation is that with increasing λ (i.e., decreasing threshold σ), we count more non-salient codewords into the histogram feature, and the final performance improves gradually. It reveals that the non-salient regions in image scenes are also very useful for classification.

The proposed FBC has a substantial superiority in computational speed compared to the state-of-the-art methods since it requires no complex feature coding strategy and even no training procedures. However, it is noted that some state-of-the-art methods outperform FBC by an obvious margin. In future works, we plan to improve FBC by elaborating more effective learning schemes jointly capturing local and global information.

7. Conclusions

This paper presents a fast, but effective method, called FBC, for extracting global feature representations of remote sensing image scenes in a straightforward way. In the FBC pipeline, we introduce unsupervised learning techniques to automatically learn a set of optimal filters from large quantities of randomly-sampled image patches, and through binarizing the feature responses, we can readily compute the feature representations for image scenes in a computationally-efficient way. The overall feature extraction stage is free of any hand-crafted features and can be computed straightforwardly. The number of filters and filter size are two important parameters in FBC, and we conclude that relatively more filters with a moderate size will result in better performance. Filters learned via different ways also lead to much different classification performance, and it is surprising that we can even achieve a promising accuracy of 83.45% on the UCM scene dataset with random filters, which are simply obtained from a normal distribution. Extensive experiments show that the FBC-based classification model can achieve encouraging performance and at the same time save much computation cost compared to the classic BOW scene classification models and newly-published UFL-based models. In addition, we also show that on the basis of FBC, spatial co-occurrence matrices and the saliency maps can improve performance to some extent. On the whole, the FBC is a promising method for scene classification in terms of accuracy and computational cost, and it can be a practical tool in some industrial scene analysis systems.

Acknowledgments: This research is supported by the National Natural Science Foundation of China under Contract No. 91338113 and No.41501462.

Author Contributions: Fan Hu and Gui-Song Xia had the original idea for the study, supervised the research and contributed to the article's organization. Jingwen Hu contributed to the part of the extension of FBC and corresponding experiments. Yanfei Zhong and Kan Xu contributed to the discussion of the design. Fan Hu drafted the manuscript, which was revised by all authors. All authors read and approved the submitted manuscript.

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

References

1. Li, D.; Zhang, L.; Xia, G.S. Automatic analysis and mining of remote sensing big data. *Acta Geod. Cartogr. Sin.* **2014**, *43*, 1211–1216.
2. Rathore, M.M.U.; Paul, A.; Ahmad, A.; Chen, B.W.; Huang, B.; Ji, W. Real-Time Big Data Analytical Architecture for Remote Sensing Application. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 4610–4621.
3. Christophe, E.; Michel, J.; Inglada, J. Remote Sensing Processing: From Multicore to GPU. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2011**, *4*, 643–652.
4. Xia, G.S.; Yang, W.; Delon, J.; Gousseau, Y.; Sun, H.; Maitre, H. Structrual High-Resolution Satellite Image Indexing. In Proceedings of the ISPRS, TC VII Symposium (Part A): 100 Years ISPRS—Advancing Remote Sensing Science, Vienna, Austria, 5–7 July 2010.
5. Lienou, M.; Maitre, H.; Datcu, M. Semantic annotation of satellite images using latent dirichlet allocation. *IEEE Geosci. Remote Sens. Lett.* **2010**, *7*, 28–32.
6. Cheriyadat, A. Unsupervised Feature Learning for Aerial Scene Classification. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 439–451.
7. Yang, Y.; Newsam, S. Spatial pyramid co-occurrence for image classification. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 1465–1472.
8. Shao, W.; Yang, W.; Xia, G.S. Extreme value theory-based calibration for multiple feature fusion in high-resolution satellite scene classification. *Int. J. Remote Sens.* **2013**, *34*, 8588–8602.

9. Chen, S.; Tian, Y. Pyramid of Spatial Relatons for Scene-Level Land Use Classification. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 1947–1957.
10. Hu, F.; Xia, G.S.; Hu, J.; Zhang, L. Transferring Deep Convolutional Neural Networks for the Scene Classification of High-Resolution Remote Sensing Imagery. *Remote Sens.* **2015**, *7*, 14680–14707.
11. Hu, J.; Xia, G.S.; Hu, F.; Zhang, L. A Comparative Study of Sampling Analysis in the Scene Classification of Optical High-Spatial Resolution Remote Sensing Imagery. *Remote Sens.* **2015**, *7*, 14988–15013.
12. Cheng, G.; Han, J.; Zhou, P.; Guo, L. Multi-class geospatial object detection and geographic image classification based on collection of part detectors. *ISPRS J. Photogramm. Remote Sens.* **2014**, *98*, 119–132.
13. Yang, W.; Yin, X.; Xia, G.S. Learning High-level Features for Satellite Image Classification With Limited Labeled Samples. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 4472–4482.
14. Xia, G.S.; Wang, Z.; Xiong, C.; Zhang, L. Accurate annotation of remote sensing images via active spectral clustering with little expert knowledge. *Remote Sens.* **2015**, *7*, 15014–15045.
15. Yu, H.; Yang, W.; Xia, G.S.; Liu, G. A color-texture-structure descriptor for high-resolution satellite image classification. *Remote Sens.* **2016**, *8*, 259, doi:10.3390/rs8030259.
16. Zhang, Z.; Huang, K.; Wang, Y.; Li, M. View independent object classification by exploring scene consistency information for traffic scene surveillance. *Neurocomputing* **2013**, *99*, 250–260.
17. Xu, Y.; Huang, B. Spatial and temporal classification of synthetic satellite imagery: Land cover mapping and accuracy validation. *Geo-Spatial Inf. Sci.* **2014**, *17*, 1–7.
18. Sivic, J.; Zisserman, A. Video Google: A text retrieval approach to object matching in videos. In Proceedings of the Ninth IEEE International Conference on Computer Vision, Nice, France, 13–16 October 2003; pp. 1470–1477.
19. Fei-Fei, L.; Perona, P. A Bayesian hierarchical model for learning natural scene categories. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 20–25 June 2005; Volume 2, pp. 524–531.
20. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110.
21. Ojala, T.; Pietikainen, M.; Maenpaa, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 971–987.
22. Xia, G.S.; Delon, J.; Gousseau, Y. Shape-based invariant texture indexing. *Int. J. Comput. Vis.* **2010**, *88*, 382–403.
23. Xia, G.S.; Delon, J.; Gousseau, Y. Accurate junction detection and characterization in natural images. *Int. J. Comput. Vis.* **2014**, *106*, 31–56.
24. Bai, X.; Bai, S.; Zhu, Z.; Latecki, L.J. 3D shape matching via two layer coding. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 2361–2373.
25. Bai, X.; Yao, C.; Liu, W. Strokelets: A learned multi-scale mid-level representation for scene text recognition. *IEEE Trans. Image Process.* **2016**, *25*, 2789–2802.
26. Shi, B.; Bai, X.; Yao, C. Script identification in the wild via discriminative convolutional neural network. *Pattern Recognit.* **2016**, *52*, 448–458.
27. Calonder, M.; Lepetit, V.; Strecha, C.; Fua, P. Brief: Binary robust independent elementary features. In Proceedings of the European Conference on Computer Vision, Crete, Greece, 5–11 September 2010; pp. 778–792.
28. Ojansivu, V.; Rahtu, E.; Heikkila, J. Rotation invariant local phase quantization for blur insensitive texture analysis. In Proceedings of the 19th International Conference on Pattern Recognition (ICPR), Tampa, FL, USA, 8–11 December 2008; pp. 1–4.
29. Hu, F.; Wang, Z.; Xia, G.S.; Luo, B.; Zhang, L. Fast binary coding for satellite image scene classification. In Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, Italy, 26–31 July 2015; pp. 517–520.
30. Lazebnik, S.; Schmid, C.; Ponce, J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Washington, DC, USA, 17–22 June 2006; pp. 2169–2178.
31. Yang, J.; Jiang, Y.G.; Hauptmann, A.G.; Ngo, C.W. Evaluating Bag-of-visual-words Representations in Scene Classification. In Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval, Augsburg, Germany, 24–29 September 2007; pp. 197–206.

32. Yang, Y.; Newsam, S. Bag-of-visual-words and spatial extensions for land-use classification. In Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, San Jose, CA, USA, 3–5 November 2010; pp. 270–279.
33. Fernando, B.; Fromont, E.; Tuytelaars, T. Mining Mid-level Features for Image Classification. *Int. J. Comput. Vis.* **2014**, *108*, 186–203.
34. Van Gemert, J.; Veenman, C.; Smeulders, A.; Geusebroek, J.M. Visual Word Ambiguity. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1271–1283.
35. Bolovinou, A.; Pratikakis, I.; Perantonis, S. Bag of spatio-visual words for context inference in scene classification. *Pattern Recognit.* **2013**, *46*, 1039–1053.
36. Kannala, J.; Rahtu, E. Bsif: Binarized statistical image features. In Proceedings of the 21st International Conference on Pattern Recognition (ICPR), Tsukuba, Japan, 11–15 November 2012; pp. 1363–1366.
37. Coates, A.; Ng, A.; Lee, H. An analysis of single-layer networks in unsupervised feature learning. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 215–223.
38. Bengio, Y.; Lamblin, P.; Popovici, D.; Larochelle, H. Greedy layer-wise training of deep networks. *Adv. Neural Inf. Process. Syst.* **2007**, *19*, 153.
39. Zhang, F.; Du, B.; Zhang, L. Saliency-Guided Unsupervised Feature Learning for Scene Classification. *IEEE Trans. Geosci. Remote Sens.* **2015**, *53*, 2175–2184.
40. Hu, F.; Xia, G.S.; Wang, Z.; Huang, X.; Zhang, L.; Sun, H. Unsupervised Feature Learning Via Spectral Clustering of Multidimensional Patches for Remotely Sensed Scene Classification. *IEEE J. Sel. Top Appl. Earth Obs. Remote Sens.* **2015**, *8*, 2015–2030.
41. Heaviside Step Function. Available online: https://en.wikipedia.org/wiki/Heaviside_step_function (accessed on 21 January 2016).
42. Pati, Y.C.; Rezaiifar, R.; Krishnaprasad, P. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In Proceedings of the Twenty-Seventh Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 1–3 November 1993; pp. 40–44.
43. Olshausen, B.A.; Field, D.J. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* **1996**, *381*, 607–609.
44. Mairal, J.; Bach, F.; Ponce, J.; Sapiro, G. Online Dictionary Learning for Sparse Coding. In Proceedings of the International Conference on Machine Learning, Montreal, QC, Canada, 14–18 June 2009; pp. 689–696.
45. Lee, D.D.; Seung, H.S. Algorithms for non-negative matrix factorization. *Adv. Neural Inf. Process. Syst.* **2001**, *13*, 556–562.
46. He, X.; Niyogi, P. Locality preserving projections. *Adv. Neural Inf. Process. Syst.* **2004**, *16*, 153–160.
47. Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* **2010**, *9999*, 3371–3408.
48. Hyvärinen, A.; Oja, E. Independent component analysis: Algorithms and applications. *Neural Netw.* **2000**, *13*, 411–430.
49. Liu, L.; Fieguth, P. Texture classification from random features. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 574–586.
50. Siagian, C.; Itti, L. Rapid biologically-inspired scene classification using features shared with visual attention. *Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 300–312.
51. Sharma, G.; Jurie, F.; Schmid, C. Discriminative spatial saliency for image classification. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3506–3513.
52. Goferman, S.; Zelnik-Manor, L.; Tal, A. Context-aware saliency detection. In Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010, 2010; pp. 2376–2383.
53. Achanta, R.; Estrada, F.; Wils, P.; Süsstrunk, S. Salient region detection and segmentation. In Proceedings of the International Conference on Computer Vision Systems, Santorini, Greece, 12–15 May 2008; pp. 66–75.
54. Bruce, N.D.; Tsotsos, J.K. Saliency, attention, and visual search: An information theoretic approach. *J. Vis.* **2009**, *9*, 1–24.

55. Jiang, H.; Wang, J.; Yuan, Z.; Liu, T.; Zheng, N.; Li, S. Automatic salient object segmentation based on context and shape prior. *Br. Mach. Vis. Conf.* **2011**, *6*, 1–12.
56. Jiang, H.; Wang, J.; Yuan, Z.; Wu, Y.; Zheng, N.; Li, S. Salient object detection: A discriminative regional feature integration approach. In Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2083–2090.
57. Achanta, R.; Hemami, S.; Estrada, F.; Susstrunk, S. Frequency-tuned salient region detection. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–26 June 2009; pp. 1597–1604.
58. Harel, J.; Koch, C.; Perona, P. Graph-based visual saliency. *Adv. Neural Inf. Process. Syst.* **2007**, *19*, 545–552.
59. Murray, N.; Vanrell, M.; Otazu, X.; Parraga, C.A. Saliency estimation using a non-parametric low-level vision model. In Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 20–25 June 2011; pp. 433–440.
60. Shen, X.; Wu, Y. A unified approach to salient object detection via low rank matrix recovery. In Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 853–860.
61. Achanta, R.; Susstrunk, S. Saliency detection using maximum symmetric surround. In Proceedings of the 2010 IEEE International Conference on Image Processing, Hong Kong, China, 26–29 September 2010; pp. 2653–2656.
62. Riche, N.; Mancas, M.; Duvinage, M.; Mibulumukini, M.; Gosselin, B.; Dutoit, T. Rare2012: A multi-scale rarity-based saliency detection with its comparative statistical analysis. *Signal Process. Image Commun.* **2013**, *28*, 642–658.
63. Rahtu, E.; Kannala, J.; Salo, M.; Heikkilä, J. Segmenting salient objects from images and videos. In Proceedings of the European Conference on Computer Vision, Crete, Greece, 5–11 September 2010; pp. 366–379.
64. Seo, H.J.; Milanfar, P. Static and space-time visual saliency detection by self-resemblance. *J. Vis.* **2009**, *9*, 1–27.
65. Hou, X.; Zhang, L. Saliency detection: A spectral residual approach. In Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, 17–22 June 2007; pp. 1–8.
66. Zhang, L.; Tong, M.H.; Marks, T.K.; Shan, H.; Cottrell, G.W. SUN: A Bayesian framework for saliency using natural statistics. *J. Vis.* **2008**, *8*, 1–20.
67. Chang, C.C.; Lin, C.J. LIBSVM: A Library for Support Vector Machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 1–27.
68. Vedaldi, A.; Fulkerson, B. VLFeat: An Open and Portable Library of Computer Vision Algorithms, 2008. Available online: http://www.vlfeat.org/ (accessed on 21 January 2016).
69. DR Toolbox. Available online: https://lvdmaaten.github.io/drtoolbox/ (accessed on 21 January 2016).
70. PCA and ICA Package. Available online: http://www.mathworks.com/matlabcentral/fileexchange/38300-pca-and-ica-package (accessed on 21 January 2016).
71. Cui, S.; Schwarz, G.; Datcu, M. Remote Sensing Image Classification: No Features, No Clustering. *IEEE J. Sel. Top Appl. Earth Obs. Remote Sens.* **2015**, *8*, 5158–5170.
72. Varma, M.; Zisserman, A. Texture classification: Are filter banks necessary? In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Madison, WI, USA, 18–20 June 2003; Volume 2.
73. Leung, T.; Malik, J. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vis.* **2001**, *43*, 29–44.
74. Schmid, C. Constructing models for content-based image retrieval. In Proceedings of the IEEE Conference on Computer Vision and Pattern, Kauai, HI, USA, 8–14 December 2001; Volume 2, pp. 39–45.