Remote Sensing Image Classification: No Features, No Clustering

Shiyong Cui, Gottfried Schwarz, Mihai Datcu, *Fellow, IEEE*

***Abstract*—In this paper, we consider the problem of remote sensing image classification, in which feature extraction and feature coding are critical steps. Various feature extraction methods aim at an abstract and discriminative image representation. Most of them are either theoretically too complex or practically infeasible to compute for large datasets. Motivated by this observation, we propose a simple yet efficient feature extraction method within the Bag-of-Words (BoW) framework. It has two main innovations. Firstly and most interestingly, this method does not need any complex local feature extraction; instead, it uses directly the pixel values from a local window as low level features. Secondly, in contrast to many unsupervised feature learning methods, a random dictionary is applied to feature space quantization. The advantage of a random dictionary is that it does not need the time-consuming process of dictionary learning yet without a significant loss of classification accuracy. These two novel improvements over state-of-the-art methods significantly reduce the computational time and enable it scalable to a large data volume. An extensive experimental evaluation has been performed and compared with other feature extraction methods. It is demonstrated that our feature extraction method is quite competitive and can achieve rather promising performance figures for both optical and SAR satellite images.**

***Index Terms*—Bag-of-words (BoW), Dictionary learning, Feature extraction, Image classification, Unsupervised feature learning.**

I. Introduction

THE exponential growth in the amount of data in various fields has given rise to the era of Big Data. Every day, 2.5 quintillion bytes of data are created and 90 percent of the data in the world today were produced within the past two years [1]. The data volume is certainly beyond the capabilities of users and systems to access the information content of the data. The fundamental challenge is to explore the large volume of data and the extraction of useful information.

In the context of earth observation, remote sensing image classification plays an important role, which since years has been an active field of research. In image classification, feature extraction is a critical step. Traditionally, feature extraction methods are being selected manually, based on domain knowledge. In this paper, we focus on feature extraction for optical and SAR satellite image classification. First, we give a brief survey of related work about image feature extraction.

From the beginning of the twenty-first century, prominent advances in texton and local feature extraction have been witnessed, leading to the Bag-of-Words (BoW) method for feature extraction. Based on previous research results, the theory of texton states that textures can be characterized efficiently by filter responses and texton distributions. Current texton theory is able to find a theoretically sound image description based on unsupervised learning of filter responses. Some filter banks, such as the Leung-Malik [2] and the Maximum Response filter set, can be applied to texture analysis using a texton learned by clustering. A texton histogram can be generated for texture description of an image. Thus, the most important problem is to select a good filter bank. For example, 48 filters were proposed by [2], which are the first and second order derivatives of a set of Gaussian filters with 6 orientations and 4 scales, 8 Laplacian of Gaussian filters, and 4 Gaussian filters. 38 filter banks were developed in [3], which include a Gaussian filter, a Laplacian of Gaussian filter, edge filters with 6 orientations and 3 scales, and a set of bar filters with 6 orientations and 3 scales.

Although various filter banks have played an important role in texture analysis, they were challenged by some patch-based methods [4], [5]. A prominent patch-based texture synthesis was proposed in [4]. Here, texture is synthesized by stitching together a number of well selected patches. The synthesized textures are far superior to the ones obtained by filter banks. The idea behind this method is that the local image patches contain enough information about the texton. Thus, this method had a profound influence on texture synthesis.

Besides patch-based methods, local feature descriptors have received a lot of attention since the invention of the scale invariant feature transform (SIFT) [6]. Local discriminative features have promoted obvious advances in image matching and object recognition. Currently, the SIFT detector is one of the most widely used methods for feature detection, which detects sparsely distributed key points for local feature extraction. Partially inspired by SIFT, the Speeded Up Robust Feature detector (SURF) [7] was proposed, which can be computed faster and is more robust against image transforms. In addition, there are some publications [8], [9] showing that dense sampling or even random sampling are able to achieve better performance than the SIFT detector as long as the number of patches is sufficiently high [8], [9]. Different sampling strategies are compared in [10] and it is concluded that a simple variance-based point selection method can be more effective than regular grid sampling, random sampling, or SIFT. Many local feature descriptors have been proposed, like SPIN Image, Rotation Invariant Feature Transform (RIFT) proposed in [11], Census Transform Histogram (CENTRIST) [12], and Local Binary Pattern (LBP) [13], [14].

Inspired by the texton image representation and the discrimination power of local features, the BoW technique was

The authors are with the Remote Sensing Technology Institute (IMF), German Aerospace Center (DLR), Oberpfaffenhofen, Germany, D-82234. E-mail: shiyong.cui@dlr.de, gottfried.schwarz@dlr.de, mihai.datcu@dlr.de

proposed in [15] for video search. Since then, within this framework, a large variety of methods have been proposed for solving various problems, for instance, image classification, image retrieval, and object recognition. The BoW technique has been recently introduced also to the remote sensing community for image annotation [16], object classification [17], target detection [18] and land use classification [19] and it has already proven its discrimination power in image classification. The original BoW framework consists of four main components: feature detection, local feature extraction, dictionary learning, and word assignment. Later it was extended to five components; feature coding was included after dictionary learning. All elements have been investigated with a lot of effort. In the BoW method, a dictionary (or codebook) is usually learned by clustering. Based on the codebook, the feature space is quantized using a nearest neighbor assignment. Two problems appear in this step: the first one is that vector quantization loses some information; the second one is that nearest neighbor feature assignment is problematic in the case of equal distances. To reduce the information loss, supervised codebook learning [20] was proposed and a Gaussian mixture model was applied to it [21]. As for feature assignment, there are two kinds of methods: hard assignment and soft assignment. Hard assignment is to assign a feature vector to its nearest element in the dictionary. However, it has been claimed that soft assignment [22] can improve the accuracy by assigning multiple code words with weighting. In order to reduce the loss of spatial information in the BoW method, the Spatial Pyramid Match (SPM) was developed by [23] to incorporate weak spatial information. In this method, an image is divided into a series of multi-scale patches where a patch comprises a window of 3×3, 6×6, or 12×12 pixels and a local word histogram of each region is computed. Then all the word histograms from all regions are concatenated to form a vector representation of the image.

Recently, sparse coding [24] instead of vector quantization has been applied to dictionary learning, which is claimed to give better performance for image classification. It is an iterative algorithm alternating between dictionary learning and sparse decomposition. The algorithm includes feature pooling, such as max, sum, and average pooling, to compute the image representation. A theoretical analysis of feature pooling is given in [25]. Unfortunately, sparse coding is computationally expensive. Based on the observation that non-zero coefficients are often assigned to nearby elements in the dictionary, Locality-constrained Linear Coding (LLC) was proposed by [26] and [27]. In contrast to soft feature assignment, LLC assumes that a feature point for coding can be reconstructed using its k nearest neighbors in the dictionary. The reconstruction coefficients can be computed by solving a least squares problem. The weights for the remaining clusters are set to zero. Therefore, sparsity is replaced with locality.

As the BoW feature vector is an intermediate feature depending on low level features, distinctive local features should be carefully designed. In general, local rotation-invariant features are preferable for image classification. Pixel values in a local patch instead of the filter responses are proposed for texture classification in [28] claiming that compact local patches can achieve better performance than a texton distribution of the filter responses. Based on this work, a random projection [29] was applied to reduce the dimension of the local feature vectors and a significant improvement in classification accuracy was shown. However, it was observed that the random projection of the local features is not rotation-invariant; thus, a sorted random projection of five local features [30], [31] was developed by the same authors, who claimed to achieve significant improvements compared with the the method of [29].

In this paper, we do not propose a universal method that works well for all kinds of data. Instead, we present a simple yet efficient method thin the Bag-of-Words (BoW) framework that can achieve very good accuracy for remote sensing image classification. This method does not need any complex local feature extraction and any time-consuming unsupervised method for dictionary learning. We show that very small patches have sufficient information for classification and challenge the role of dictionary learning. A random dictionary gives superior performances in our cases. The major contributions of this paper are as follows:

- Without any complex local feature extraction, we use the pixel values in a very compact local neighborhood,e.g., taken from a 3×3 window and a column-wise conversion into a vector of elements (“Vectorized Patch”), as low level features for the BoW method.
- Instead of unsupervised dictionary learning, we randomly select some feature points and use them as our dictionary.
- We did an extensive evaluation of the effects of different parameters in BoW and drew a comparison among several options. All the questions listed in Section II-B could be clearly answered experimentally.

The rest of the paper is organized as follows. In Section II, we first present the framework of BoW feature extraction and then, we propose our methods for feature extraction and dictionary learning. In Section III, we present three datasets that have been used for evaluation. A detailed evaluation and comparison on the SAR dataset is given in Section IV. Experimental results using an open optical dataset is shown in Section V. Finally, a conclusion is drawn in Section VI.

II. BoW Feature Extraction

In this section, we first present the general framework of BoW feature extraction and then, we propose our method for feature extraction and dictionary learning.

A. BoW Feature Extraction Framework

The framework of BoW feature extraction shown Fig. 1 is composed of five steps, which are patch sampling, local feature extraction, dictionary learning, feature coding, and feature pooling. Assume we have a dataset of N images $I_i, i = 1, ..., N$, the first step is to sample a collection of patches from the images in the database. This can be done by dense sampling or sparse detection. The second step is to extract local descriptor vectors $\mathbf{x}_i^j \in \mathbb{R}^D, j = 1, ..., M$ from all patches. The third one is learning a dictionary $\mathbf{D} = (\mathbf{d}_1, ..., \mathbf{d}_K) \in \mathbb{R}^{D \times K}$ with K words using all local features.

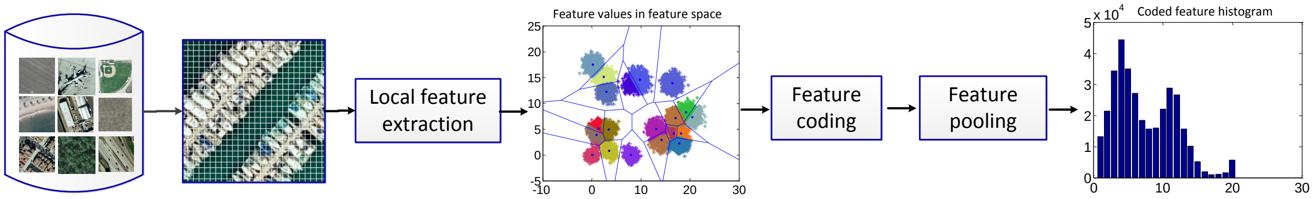


Fig. 1. The framework of the Bag-of-Words model consists of five steps: patch sampling, local feature extraction, dictionary learning, feature coding, and feature pooling.

Normally, this is done by a time consuming unsupervised learning method, such as k-means clustering or a Gaussian mixture model. The elements $\mathbf{d}_i$ in a dictionary are the centers of the clusters. The next step is to find a dictionary-based representation $\mathbf{v} = [v_1, ..., v_K]$ for each previously extracted local descriptor $\mathbf{x}$. This can be done using hard feature assignment or soft assignment. Hard assignment assigns a single label, i.e., the index of the nearest neighbor in the dictionary, to each local descriptor $\mathbf{x}$. Formally, it is defined as:

$$v_k(\mathbf{x}) = \begin{cases} 1 & \text{if } k = \text{argmin}_i \|\mathbf{x} - \mathbf{d}_i\|^2 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

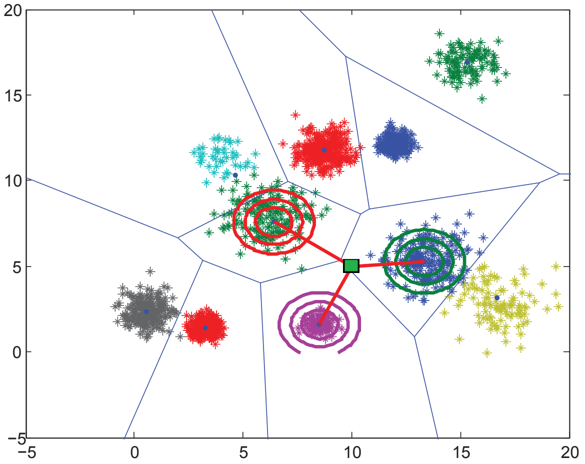


Fig. 2. Soft feature assignment using a kernel codebook: A local descriptor denoted by a green square is assigned to three nearest neighbors. So the final coded feature vector of this local descriptor has three non-zero values. In contrast, hard assignment attributes a local descriptor only to its nearest neighbor.

Thus, the final descriptor representation $\mathbf{v} = [v_1, ..., v_K]$ has only one non-zero element. The last step is to do the sum-pooling [1] of all local descriptors extracted from one image $\mathbf{v}_i = \text{sum}(\mathbf{v}_i^j, ..., \mathbf{v}_i^j)$. In contrast, soft assignment tries to assign a descriptor to multiple elements in the dictionary by proportionally weighting the distances to the nearest neighbors (cf. Fig. 2). Mathematically, the final descriptors are computed as follows:

$$v_k(\mathbf{x}) = \frac{\exp\left(\|\mathbf{x} - \mathbf{d}_k\|_2^2/\sigma\right)}{\sum_{k=1}^{P} \exp\left(\|\mathbf{x} - \mathbf{d}_k\|_2^2/\sigma\right)} \tag{2}$$

where $\mathbf{d}_k, k = 1, .., P$ are the P nearest neighbors of a local descriptor $\mathbf{x}$ in the dictionary and σ is the smoothing parameter of a kernel function.

B. Methodology

During our investigations, we encountered a number of relevant questions that had to be addressed. The most important potential problem areas are summarized below:

1) What are the best local patch size and the best patch sampling strategy?
2) What local features should be extracted?
3) What is the strategy of dictionary learning within the BoW framework? A universal dictionary or multiple class-specific dictionaries?
4) Involved sparse coding or simple vector quantization?
5) Nearest neighbor assignment or multiple assignments?

All these problems have to be tackled with care. Detail analysis of critical parameters is very important for a well understanding of the BoW method; this has been observed in previous work. If we optimize all these components, a simple unsupervised feature learning algorithm could be able to achieve state-of-the-art accuracy. This has been observed by Coates and Ng in [32]. In the following, we try to circumvent a joint overall optimization and we decompose the overall optimization into individual steps that can give us some hints about what causes some algorithms to perform well and others to perform poorly.

The first problems are the patch size and the patch sampling strategy, which are practically related. If the patch size is quite large, the dimensionality of the local features is very high [29], which makes a subsequent unsupervised dictionary learning time-consuming, and thus infeasible for large scale databases. In addition, there would be large overlaps among patches if the patch size is large. This could potentially degrade the feature space. As for the patch sampling strategy, we will compare regular dense sampling and random sampling in Section IV.

Another important problem is what local features can be extracted with minimum computational effort. There are many local features that have been proposed, as outlined in Section I. We analyzed several discriminative local features and will show that the vectorized pixel values of very small patches, e.g., defined by windows of 3 × 3 pixels, provide enough information for discrimination. We demonstrate that this simple local feature vector can achieve a rather promising performance for image classification. The main advantages are its simplicity and the low computational cost.

[1] Sum-pooling is equivalent to computing the histogram in the case of hard feature assignment.

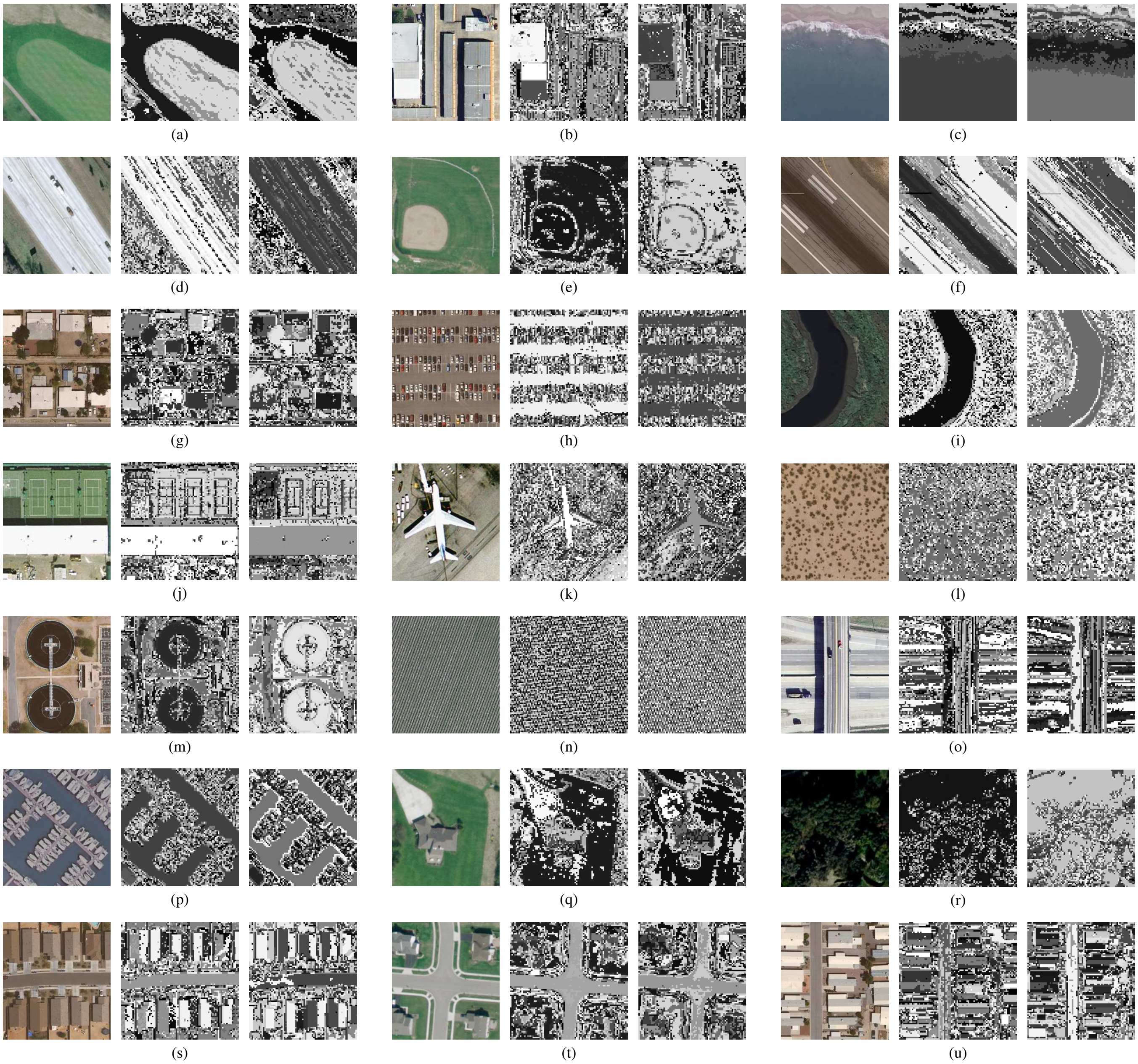


Fig. 3. Comparison of vector quantization using a random dictionary and k-means clustering on the UC Merced Land Use Dataset (see Section III-B). Three examples are given for each of the 21 classes. The first color image is an example from each class. The second and third images in each group are the dictionary entries using a random dictionary and a k-means dictionary. Both dictionaries have the same size of 200 entries. The semantic labels are given in Section III-B.

The third problem is about the role of the dictionary that is usually learned by various unsupervised clustering algorithms. Dictionary learning is always taken for granted. However, in the BoW method, this step is most time consuming. In the case of large datasets, it is prohibitively time consuming to learn a dictionary. The goal of dictionary learning is to find a universal reference for feature coding. This universal reference is not necessarily coincident with the actual cluster centers. We show that a random dictionary, collected by a random selection of some local descriptors in the feature space, is similar to one that is carefully learned by an unsupervised clustering method. Typical examples are given in Fig. 3. Here, we use the pixel values of a 3×3 vectorized patch as a local feature vector; the patches are sampled regularly from the given images. Then we compare the results of vector quantization using k-means with the results of a random dictionary. From the results of vector quantization, we see that a random dictionary can achieve similar performance as k-means. For the purpose of quantitative analysis, the vector quantization errors using both k-means and a random dictionary with the same size of 200 entries are shown in Fig. 4. Although the quantization error using a random dictionary is larger than for k-means, the computational cost is significantly reduced without incurring a loss in classification accuracy (see Section IV-F). Thus, the

```
**Data**: A database of images $I_i, i = 1, 2, N$ and the size of dictionary $K$.
**Result**: Feature vector of all images $I_i$ $Feat\_Matrix(K, N)$ in the database.
// initialization ;
$D =$ zeros$(9, K), IDs = [\,], nb\_patches =$ zeros$(1, N), idx = 1;;$
for $i \leftarrow 1$ to $N$ do
    // compute the number of patches ;
    $nb\_patches(i) \leftarrow$ CompNbPatches$(I_i)$ ;
    $IDs \leftarrow [IDs \quad 1 : nb\_patches(i)]$ ;
    $idx \leftarrow idx + nb\_patches(i) - 1$ ;
end
// sampling a random dictionary;
$randIDs \leftarrow$ randperm$(idx)$ ;
$dictIDs \leftarrow randIDs(1 : K)$ ;
$cum\_nb\_patches \leftarrow$ cumsum$(nb\_patches)$ ;
for $i \leftarrow 1$ to $K$ do
    $ID \leftarrow dictIDs(i)$ ;
    for $j \leftarrow 1$ to $N$ do
        if $cum\_nb\_patches(j) > ID$ then
            $im\_id \leftarrow j$ ;
            break;
        end
    end
    $D(:, i) \leftarrow$ readPatch$(I_{im\_id}, IDs(ID))$ ;
end
// loop over all the images and extract feature ;
for $i \leftarrow 1$ to $N$ do
    $patches \leftarrow$ im2patches$(I_i)$ ;
    $assign \leftarrow$ zeros$(1, nb\_patches(i))$ ;
    for $j \leftarrow 1$ to $nb\_patches(i)$ do
        $k \leftarrow$ NearestNeighbor$(patches(:, j), D)$ ;
        $assign(j) \leftarrow k$ ;
    end
    $fv \leftarrow zeros(1, K)$ ;
    for $j \leftarrow 1$ to $nb\_patches(i)$ do
        $fv(assign(j)) \leftarrow fv(assign(j)) + 1$ ;
    end
    $Feat\_Matrix(:, i) = fv$ ;
end
```

Algorithm 1: The proposed algorithm.

final feature vectors of an image are similar. This point is pivotal, because time consuming clustering is avoided. Thus, it makes BoW applicable and scalable for large databases. Similar observations have been presented by Coates and Ng in [33]. Another advantage of this method is that we do not have to load all the features into memory. Only the random dictionary is needed to be loaded into memory. Thus, the memory requirements are significantly reduced. This is very important for large datasets because they probably will not fit into memory in many cases. The entire algorithm is summarized in Alg. 1. In this pseudo code, $Feat\_Matrix$ is the final feature matrix, in which each column are the BoW features; $CompNbPatches(I_i)$ is a function to calculate the number of patches that can be sampled from the image I_i, $cumsum(\mathbf{x})$ is a function to compute the cumulative sum of vector $\mathbf{x}$.

The last two problems involve feature coding methods, which try to make the features more discriminative. There are a number of different feature coding methods [34]. Although a comparison has been done in [34], they are compared using photos, not remote sensing images. Remote sensing images are quite different from photos, such as no discrimination between background and foreground. In addition, given a remote sensing image, it is hard to tell about the what objects present in the image. Thus, we compare vector quantization and sparse coding as well as nearest neighbor assignment and multiple assignments in Section IV.

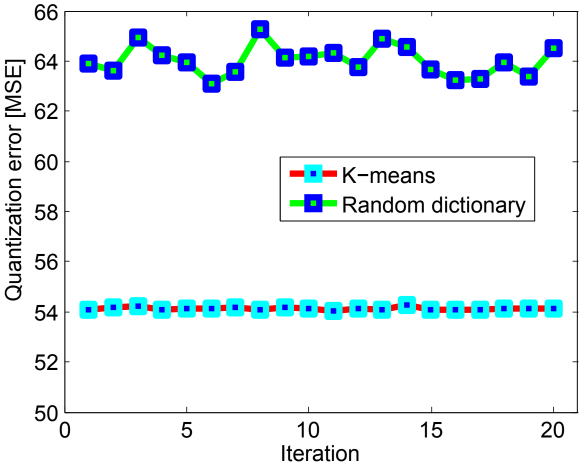


Fig. 4. Vector quantization error of a k-means and of a random dictionary after 20 test runs of k-means clustering. The average distance of all feature vectors to their nearest neighbors is computed as a measure of the quantization error.

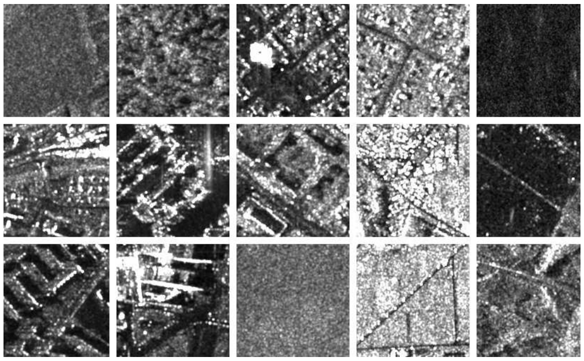


Fig. 5. Example images of 160 × 160 pixels from 15 classes of 3434 TerraSAR-X images being used for evaluation. They comprise 7 classes derived from urban areas. The number of images in each class range form 118 to 430.

III. Datasets

For a detailed quantitative evaluation of the proposed options, two databases were prepared. The first one is a database of space-borne SAR images, while the second one contains optical satellite images. These two databases are introduced in the following sections.

A. SAR Images

The first database is composed of 15 classes of altogether 3434 TerraSAR-X sub-scenes [35] with a size of 160 × 160 pixels and a pixel spacing of about 3 m (cf. Fig. 5). These 15 class are representative classes that are often seen from remote sensing images. The sub-scenes are cut from radiometrically enhanced high resolution Stripmap TerraSAR-X images with good signal-to-noise ratios. This dataset was interactively compiled from 100 TerraSAR-X images covering many countries over the world using an active learning system [36]. We could discriminate 15 classes; among them there are 7 classes of urban areas, which is sufficient to evaluate the methods for urban area classification. In addition, there are 3 classes related to agricultural fields. The remaining classes contain grassland, forest, mountain areas, railway tracks, and ocean water.

B. *Optical Images*

The second database we used is the UC Merced Land Use Dataset [19][2], which is publicly available and served as a test bed for several publications. Thus, we can easily compare our results with other methods that have been applied to this dataset. The data contain manually extracted sub-scenes from large images existing in the USGS National Map Urban Area Imagery collection covering various urban areas around the United States. The pixel spacing of this public domain imagery is 1.0 foot. The database comprises 21 classes and each class contains 100 scenes. Example scenes from each class are shown in Fig. 3. The semantic labels of the 21 classes are 'agricultural', 'airplane', 'baseball diamond', 'beach', 'buildings', 'chaparral', 'dense residential', 'forest', 'freeway', 'golf course', 'harbor', 'intersection', 'medium residential', 'mobile home park', 'overpass', 'parking lot', 'river', 'runway', 'sparse residential', 'storage tanks', and 'tennis court'. A quantitative evaluation on this dataset is described in Section V.

IV. Experiments and results on the SAR images

In this section, we investigate the five problems listed in Section II-B based on our SAR dataset. A series of experiments are performed where we evaluate the effects of varying one BoW parameter while keeping the other parameters fixed. In addition, we compare our BoW results with other methods. In all the following evaluations, 30 training samples are randomly selected from each class and used for classification training and the remaining images are used as test data. The classifier used by us is a one vs. one C-SVM [37] with a χ^2 kernel function. The parameter C is empirically set to 1000. The five problems listed in Section II-B are analyzed. The classification performance is measured in 20 test runs and we show their average accuracy.

A. *Patch Size*

In this experiment, different window sizes (from 3×3 to 21×21 pixels) are used for patch sampling and cutting. The local feature vector is a column-wise vectorization of all pixel values within the local window. Then k-means clustering is applied to learn a dictionary with a size of 200 entries. Three evaluations are carried out. In the first evaluation, we do not consider overlapping patches; thus, the number of patches decreases as the patch size increases. The classification accuracy versus patch size is shown as the red curve in Fig. 6. The number next to each point on the curve is the number of patches having been sampled from an image. Obviously, the accuracy decreases as the patch size increases. Here, the shrinking number of selectable patches may be a reason for the decreasing accuracy, rather than the patch size. Therefore, in the second evaluation we allow overlapping patches in order to increase the number of patches that can be sampled from an image. The resulting accuracy versus patch size is plotted as the green curve in Fig. 6. Similarly to the first evaluation, the accuracy decreases as the patch size increases. This is consistent with the observation in the first evaluation. However, it is worth to note that increasing the number of patches will increase the accuracy for a given patch size. Although we allow overlaps between adjacent patches, the number of patches still decreases as the patch size increases, which can be seen from the number next to the plotted points. In the last evaluation, we keep the number of patches fixed; we randomly sample a constant number of 2704 patches from each image, which is the total number of 3×3 patches in the case of no overlap. The resulting accuracy is shown as the blue curve in Fig. 6. Apparently, the accuracy still decreases as the patch size increases. In addition, the blue curve is very similar to the green curve in Fig. 6. We can see that increasing the number of large patches will not lead to improved accuracies.

Therefore, we can conclude that a compact neighborhood size of 3×3 pixels is better than a large patch size, which is not consistent with the claim by [30] that the patch size must be large enough to encompass the dominant texture variations. This is probably because the images used for evaluation are different. With a smaller patch size, the image content variation can still be captured by the word histogram in the BoW framework, because the word histogram counts the number of clusters that occur in the image. The reason that the accuracy for a large patch size is worse can be explained by the feature space that cannot be well separated. In addition, there is a large overlap between adjacent patches for large patch sizes. Although the number of patches is sufficient due to their large overlap, the patches might not be representative enough to learn clusters in the feature space. Consequently, the final word histogram has less discrimination power.

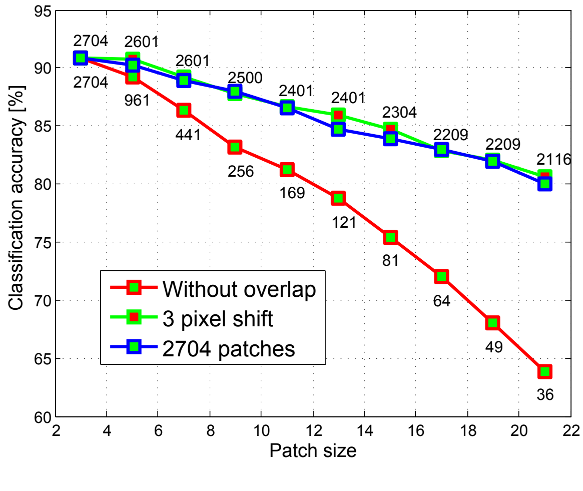
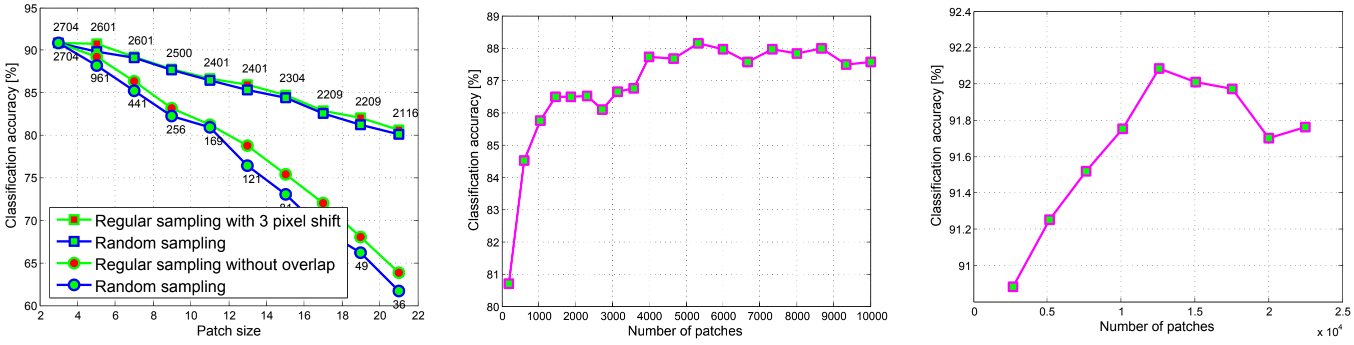


Fig. 6. Patch size evaluation: The red curve is the classification accuracy versus patch size using regular patch sampling without overlap. The number next to each point on the curve is the number of patches being used. The green curve is the case with a 3 pixel shift along both dimensions. The blue curve is the classification accuracy versus patch size using random sampling with 2704 patches.

B. *Patch Sampling Strategy*

In this experiment, we compare regular dense patch sampling (with and without overlap) with random sampling of differently sized patches while keeping the number of patches fixed. In case of random sampling, the row and column positions of the patches are determined by drawing random

[2] http://vision.ucmerced.edu/datasets/landuse.html



(a) (b) (c)

Fig. 7. Evaluation of patch sampling: (a) Comparison of regular sampling and random sampling with the same number of patches; (b) Impact of the number of patches sampled from an image with a patch size of 11 × 11 pixels; (c) Impact of the number of patches sampled from an image with a size of 3 × 3 pixels. The first abscissa point corresponds to the maximum number of 3041 patches that we can obtain by regular sampling from an image. When we increase the number of patches by irregular random sampling, the accuracy reaches a peak near 4 times the maximum number of patches for regular dense sampling, which is much less than the maximum number of patches that can be sampled from the image. This implies that increasing the number of patches in random sampling can increase the accuracy slightly; however, it is not necessary to use all patches from an image.

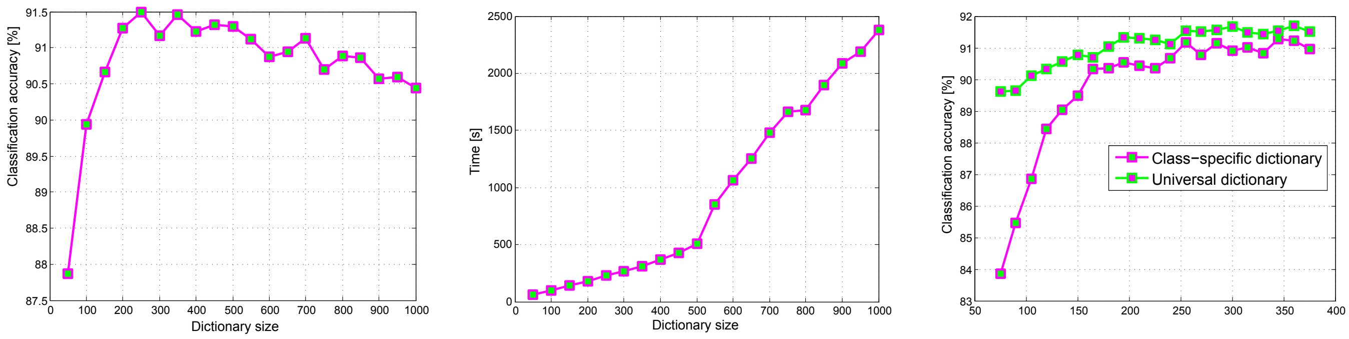
samples from a uniform distribution. The regular dense sampling with overlap is the same as the case corresponding to the green curve in Fig. 6. The number of patches is the same as in the case of regular sampling with overlap, but now the sampling strategy is replaced with random sampling. The resulting classification accuracy versus patch size is plotted as the blue curve with squares in Fig. 7(a). It can be clearly seen that, although regular dense sampling is slightly better than random sampling, there is no big difference as long as the number of patches remains the same and the entire image is fully covered. In a second evaluation, we compare random sampling with regular dense sampling without overlap while keeping the number of patches fixed. The classification accuracy versus patch size is shown as the blue curve with superimposed circles in Fig. 7(a). In this case, regular sampling is also slightly better than random sampling because, in practice, random sampling will not fully cover the entire image. To obtain a definite answer to the influence of the sampling strategy, we verify the impact of increasing the number of patches that are randomly sampled for small and large patch sizes. The effect of increasing the number of patches with random sampling is shown in Fig. 7(b) for a patch size of 11 × 11 pixels and in Fig.7(c) for 3 × 3 pixels. For a large patch size of 11 × 11 pixels, the classification accuracy initially increases as the number of patches grows. However, the attainable accuracy becomes stable beyond 4000 patches, which means there is no substantial gain in accuracy by increasing the number of patches, because a large number of patches will be duplicated, which can prohibit good clustering. On the other hand, an excessive number of patches will increase the computational burden. For smaller patches, the relationship between classification accuracy and the number of patches is shown in Fig. 7(c). It can be seen that increasing the number of small patches can lead to a slight increase in accuracy. However, even increasing the number of large patches in random sampling cannot reach the accuracy of regular dense sampling with small patches. Therefore, we conclude that two conditions have to be satisfied for patch sampling: The first one is that the entire image needs to be covered by the patches. The second one is that the number of patches has to be sufficiently high such that the statistics can be accurately estiamted. Regular dense sampling with small patches has a slightly better accuracy than random sampling with the same number of patches. For large patch sizes, increasing the number of patches with random sampling can improve the accuracy, but this approach is inferior to regular sampling with small patches, which confirms the conclusion of the previous evaluation.

C. *Dictionary Size*

The classification accuracy may also depend on the dictionary generated by k-means clustering. We tested a range of dictionary sizes to evaluate their impacts. In the case of 3×3 patches, the classification accuracy versus dictionary size is shown in Fig. 8(a). It can be clearly seen that the accuracy initially increases with a growing dictionary size because a small dictionary cannot capture the full distribution of the feature space and is thus underfitting the model. However, the accuracy reaches its peak when the dictionary size reaches around 250 entries. Beyond that point, the accuracy decreases again because the feature space suffers from overfitting by a large dictionary. In addition, the required computation time on a standard PC versus different dictionary sizes is plotted in Fig. 8(b). The time increases linearly with dictionary size. Therefore, for a practical application, one should select an appropriate dictionary size with respect to both classification accuracy and computing time. In our case, a value of 250 seems to be a good choice.

D. *Universal Dictionary or Class Specific Dictionary*

An important issue in dictionary generation is either to construct a universal dictionary for all occurring classes, or a class specific concatenated dictionary. Fig. 8(c) shows the classification accuracy versus dictionary size. It becomes evident that a universal dictionary always performs better than



(a) (b) (c)

Fig. 8. Impact of the dictionary size on the accuracy and the computational cost: (a) Dictionary size; (b) Computational cost; (c) Comparison of universal and class-specific dictionaries.

a concatenation of class specific sub-dictionaries. The reason is that different classes may have some clusters in common. The universal dictionary can capture the global feature distribution without considering the distributions of individual classes. In contrast, class specific sub-dictionaries may have some common clusters but they are considered separately; thus, the feature space is prone to underfitting. However, the computational effort to obtain a universal dictionary is much higher than generating a class specific dictionary as, in addition to the curse of dimensionality, a large volume of feature vectors is involved in the clustering.

E. Extraction of Local Features

From the previous sections, we know that the pixel values taken from a compact neighborhood can achieve very promising classification accuracy with a medium size dictionary. Therefore, we compare our vectorized patch baseline method with six other pixel sorting methods proposed in [11], [30] and [31]. These are RIFT and five Sorted Random Projection (SRP) methods that differ in how to sort the pixel values or sorted pixel differences taken from a local window, namely SRP Global, SRP Square, SRP Circular, SRP Radial-Diff, and SRP Angular-Diff. The reason why we compare our method with these pixel sorting alternatives is that they are quite competitive. In the following three evaluations, we vary one parameter while keeping the other ones fixed.

In the first evaluation, a dictionary with a size of 200 elements is learned using k-means. We evaluate the impact of local feature sorting with patch sizes ranging from 3×3 to 21×21 pixels with a 3 pixel shift in two directions. The resulting classification accuracy when using different patch sizes is shown in Fig. 9(a). It is interesting to see that a sorting of the pixel values may have a considerable impact for large patch sizes, but the improvement remains slight for smaller patch sizes. Three sorting options, namely, SRP Global, SRP Square, and SRP Circular rank on top and exhibit similar behavior when the patch size increases. In these cases, the accuracy decreases as the patch size increases, which confirms our conclusion drawn previously in Section IV-A. In contrast, the SRP Radial-Diff option shows an increasing accuracy when the patch size increases but its initial accuracy is lower than that of our vectorized patch method. SRP Radial-Diff reaches its peak accuracy for a patch size of 7×7 pixels. Beyond that point, the classification accuracy decreases again and is similar to our vectorized patch method. SRP Angular-Diff has a sharp improvement in accuracy when changing from 3×3 to 5×5 pixels. For patch sizes between 7×7 and 13×13 pixels, its performance is inferior to the baseline vectorized patch method. However, for smaller patch sizes, the baseline vectorized patch method performs much better than SRP Radial-Diff and SRP Angular-Diff. On the other hand, RIFT ranks last when compared with the other options. From this evaluation, we can conclude that using all pixel values in a patch gives good accuracies.

In the second evaluation, we use fixed patch sizes of 3×3, 5×5, and 7×7 pixels and vary the dictionary size from 50 to 500 entries. In this comparison, the baseline method is a vectorized patch of 3×3 pixels. The resulting classification accuracy versus dictionary size is shown in Fig. 9(b). Generally, the accuracy is higher for large dictionaries but stabilizes for sufficiently large dictionaries, which is consistent with the conclusion drawn previously in Section IV-C. It can be clearly seen that for a sufficiently large dictionary and a small patch size the performances of the SRP Global, SRP Square, and SRP Circular options are not much different from our baseline method. However, the baseline method performs much better than the SRP Radial-Diff and SRP Angular-Diff options for all dictionary sizes.

In the third evaluation, we vary the number of training samples while keeping fixed the dictionary size and the patch size. The accuracy of all five options when varying the number of training samples is shown in Fig. 9(c). Obviously, SRP Angular-Diff and Radial-Diff perform worse than all other four options. The common characteristic of all these four options is that they use all pixel values in their local neighborhood. Another observation is that the accuracy differences are negligible for very small patch sizes. As the patch size increases, the advantage of a sorting operation becomes obvious. However, the accuracy decreases as the patch size increases. Therefore, aiming at the best accuracy, we should use all pixel values of a very small patch as low level feature vector.

We conclude that the vectorized pixels from a very small neighborhood perform quite well in the BoW model for SAR image classification. Sorting the pixel values in the

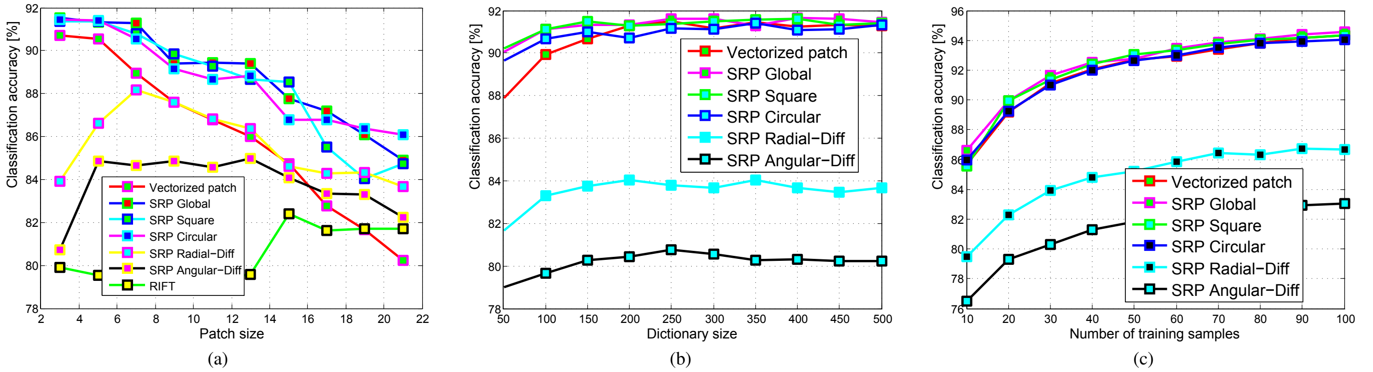


Fig. 9. Evaluation of pixel sorting options: (a) Comparison of vectorized patches with six other feature sorting options using a fixed dictionary size of 200 entries; (b) Comparison using different dictionary sizes with a patch size of 3 × 3 pixels; (c) Comparison using a different number of training samples, a patch size of 3 × 3 pixels, and a dictionary size of 200 entries.

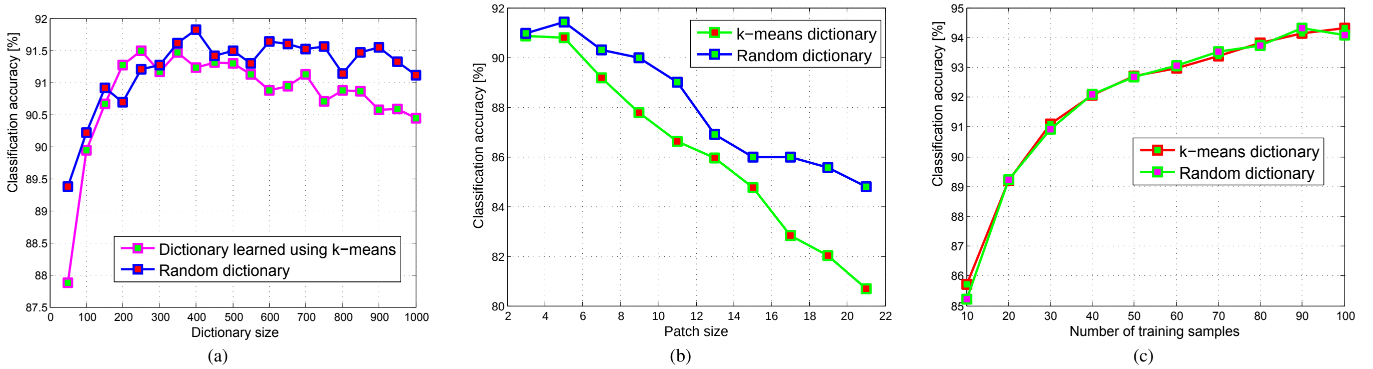


Fig. 10. Comparison of a random dictionary with a dictionary learned using k-means: (a) Comparison using different dictionary sizes; (b) Comparison using different patch sizes; (c) Comparison using different numbers of training samples.

compact patch performs slightly better but does not give much improvement for small patch sizes. However, SRP Angular-Diff and SRP Radial-Diff always perform worse than the vectorized pixels for small patches. On the other hand, in terms of computational effort, SRP Circular, SRP Angular-Diff and SRP Radial-Diff options are more time consuming than our vectorized baseline because they involve interpolation at non-integer positions. From a practical point of view, our vectorized baseline is preferable in the case of large scale applications.

F. Learned Dictionary or Random Dictionary

In this experiment, we compare random dictionary learning and k-means dictionary learning in terms of classification accuracy. Three evaluations are performed. In the first evaluation, we use the vectorized patch of a 3 × 3 pixel window as a low level feature vector. The elements in the random dictionary are randomly selected from all the local feature vectors. The classification accuracy versus dictionary size is shown in Fig. 10(a). We can clearly see that there is not much difference between a random dictionary and the one learned using k-means. In the case of large dictionaries, a random dictionary is even better. This is very important for practical applications as dictionary learning using k-means is usually quite time consuming and may become prohibitively slow for large datasets. From this evaluation, we see that it is not necessary to spend time for learning a dictionary using unsupervised learning methods. As long as the elements in the dictionary can provide full support for the data points in the feature space, even a random dictionary can give very good accuracy.

In the second evaluation, we investigate the performance of a random dictionary versus patch size. We fix the dictionary size to 200 entries and vary the patch size from 3 × 3 to 21 × 21 pixels with a 3 pixel window shift in two directions. The classification accuracy versus patch size is shown in Fig. 10(b). It becomes evident that a random dictionary is superior to dictionaries learned by k-means. In the last evaluation, we change the number of training samples while keeping fixed the patch size of 3 × 3 pixels and the dictionary size of 200 entries. The classification accuracy versus the number of training samples is shown in Fig. 10(c). We can clearly see that they are almost the same. Through these three evaluations, we conclude that a random dictionary can achieve a good performance and, in some cases, an even better accuracy than k-means.

G. Sparse Coding or Vector Quantization

In this section, we compare different feature coding methods. Vector quantization (hard feature assignment) [15] is used as a baseline for comparison with Fisher Vector (FV) [38], Kernel Codebook Encoding (KCE) [22], and Locality-constrained Linear Coding (LLC) [27]. To demonstrate the advantage of a random dictionary, its classification accuracy is also plotted as a reference in Fig. 11.

In this evaluation, the patch size used by us is 3×3 pixels and the dictionary sizes vary from 50 to 500 entries. Three local feature extractors, namely SRP Global, SRP Angular-Diff, and vectorized patches, are chosen for evaluating the feature coding methods.

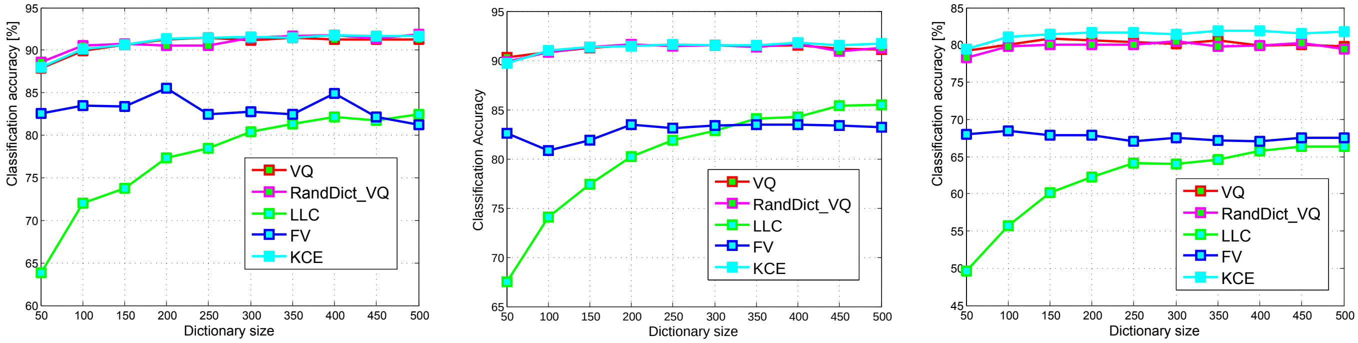
All results are shown in Fig. 11. Obviously, vector quantization using a random dictionary has very good performance. Although the use of a kernel codebook was proposed to overcome the drawback of vector quantization, its actual improvement is negligible. It is only slightly better than vector quantization for the less discriminative feature sorting option SRP Angular-Diff. Therefore, it seems that there is no gain in accuracy by assigning a local feature vector to multiple neighbors. The performances of both vector quantization and kernel codebook are quite stable with respect to the dictionary size. It is interesting to see that both LLC and FV perform worse than vector quantization. FV performs even worse but its accuracy remains quite stable versus dictionary size. The most devastating characteristic of FV is that it is very time consuming to learn a mixture model in the case of a large dictionary. The least performing method is LLC, whose performance improves with increasing dictionary size. From the accuracies shown in Fig. 11, we can see that both the local feature descriptor and the feature coding method are very important. Bad choices of these components will reduce the overall performance. We conclude that although there are many methods trying to improve vector quantization by reducing the information loss, we do not observe much gain in accuracy.

H. Comparison with Other Methods

In the last experiment, we compare the BoW method using vectorized pixels of a 3×3 patch and a random dictionary with state-of-the-art feature extraction methods, namely Gabor feature extraction, GLCM feature extraction, wavelet feature extraction, and feature extraction based on Short-Time Fourier Transform (STFT), Quadrature Mirror Filters (QMF) and Fractional Fourier Transform (frFT).

- Gabor texture features are the statistics of Gabor filter responses. A Gabor filter is characterized by its scale and orientation. We compare two sets of statistics: The first set consists of the mean and the variance of the sub-bands [39], while the second set contains the log-mean and the log-variance of the sub-bands; this combination has been demonstrated as a good choice for SAR image retrieval in [40]. The number of scales and orientations are set to 4 and 6, respectively. Thus, the dimension of the feature vector is 48.
- GLCM texture features [41] are the statistics of the so-called co-occurrence matrix, which is defined by the second order statistics of a pair of pixels being offset from each other by a given number of horizontal and vertical pixel shifts. To reduce the computational complexity, a coarse quantization of the image gray levels is usually applied prior to calculating the co-occurrence matrix. As suggested by [42], setting the number of levels to a value of less than 24 can produce unreliable classification results, while a large number of levels (greater than 64) are deemed unnecessary since they do not improve the classification accuracy and are computationally costly. Therefore, we set the number of quantization levels to 32. The number of orientations is set to 4 and the number of shifts ranges from 1 to 4. The statistics we compute are autocorrelation, contrast, correlation, cluster prominence, cluster shade, dissimilarity, energy, entropy, homogeneity, maximum probability, sum of squares, sum average, sum variance, sum entropy, difference variance, difference entropy, information measure of correlation, inverse difference, normalized inverse difference, normalized inverse difference moment [41], [42]. Then, the total dimension of the corresponding feature vector becomes $20 \times 4 \times 4 = 320$.
- Alternatively, the texture features based on a wavelet transform are the mean and variance of the filter bank responses. In our case, an image is decomposed into 3 levels using both a non-decimated 2D wavelet transform (NDWT) and a dual tree complex wavelet transform (DTCWT) [43]. Similar to the Gabor case, two sets of features are computed. In the non-decimated 2D wavelet transform, a Daubechies filter is applied, while in DTCWT, near-symmetric 13,19-tap filters are being used for the first level and Q-Shift 14,14-tap filters are employed for all higher levels. The dimensions of the two feature vectors are 18 and 36 respectively.
- The extracted STFT features [44] are 6 parameters based on a short-time Fourier transform, which include the mean and variance, the spectral centroid and the spectral flux in horizontal and vertical direction. The spectral centroid is the centroid of the short-term Fourier transform; it is a measure of spectral brightness.
- The QMF features [45] we computed are the mean and variance of all the sub-bands in the pyramid. The number of levels is set to 3, thus, the sub-band pyramid comprises 1 low pass band, 3 horizontal sub-bands, 3 vertical sub-bands, and 3 diagonal sub-bands. The corresponding feature vector contains 20 elements.
- Finally, the use of Fractional Fourier transform (frFT) features was proposed by [40] and [46] for SAR image classification. Here, the log-moment and log-variance of all sub-bands are used as a feature vector to characterize a SAR image patch. The only free frFT parameter is the number of angles, which is assumed to be 18. Therefore, the feature vector dimension is 36 elements.

The classification accuracies of our feature extraction methods (including some logarithmic versions of known methods) for all 15 SAR image classes are shown in Fig. 12. It can be clearly seen that the BoW method using vectorized patch pixels and a random dictionary performs significantly better than all other methods and has an average accuracy of more



(a) Vectorized pixels

(b) SRP Global

(c) SRP Angular-Diff

Fig. 11. Evaluation of feature coding methods using three different local feature vectors and different dictionary sizes: (a) Vectorized patch; (b) SRP Global; (c) SRP Angular-Diff.

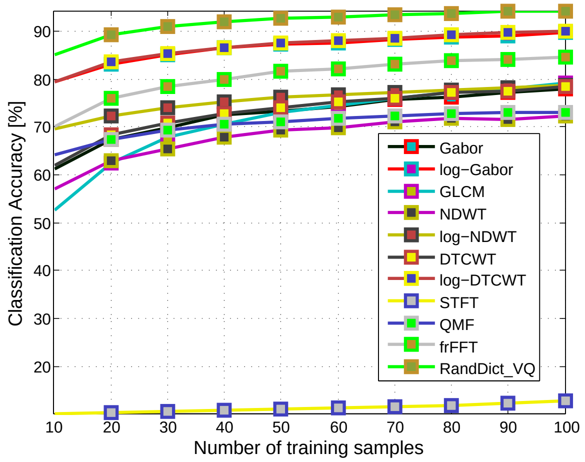


Fig. 12. Comparison of the BoW method with state-of-the-art feature extraction methods.

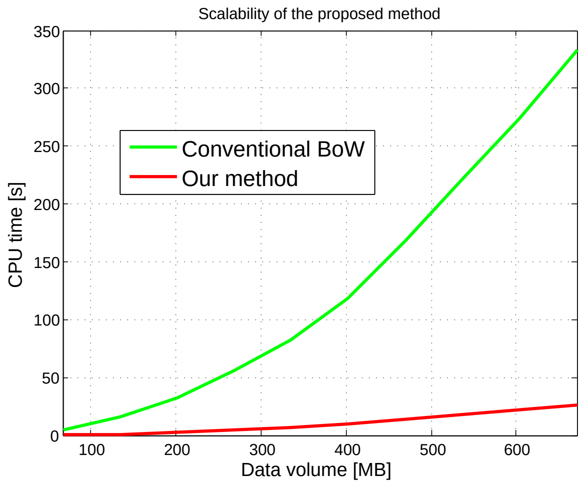


Fig. 13. Scalability of the proposed method.

than 90%. In contrast, the average accuracies of all the other methods are lower than 90%. Log-Gabor and log-DTCWT have similar performances next to BoW, followed by frFT that performs better than all the remaining methods. In addition, we can see that the logarithmic versions of Gabor, NDWT, and DTCWT perform better than their linear counterparts. The STFT method lies far behind; the reason for it could be the lower dimension of its feature vector.

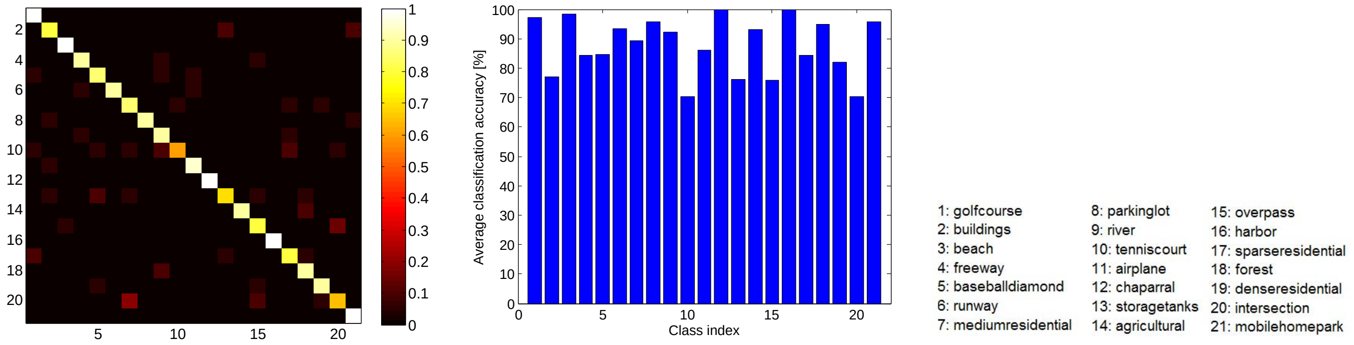
I. Scalability of the method

Since the selected data is indeed not very big, in this section we demonstrate the scalability of this method. We compare the scalability of the proposed method with conventional BoW method. To this end, we increase gradually the data volume that have been selected for feature extraction and measure the processing time. The results are shown in Fig. 13. From this figure, we can see that the proposed method performs much faster than conventional BoW and it can be easily applied to a large data without much need for processing time. In addition to the processing time, the memory consumption is significantly reduced because conventional BoW method needs to load all the data into memory in order to learn a dictionary. Thus the memory requirement is linearly increasing as the data volume increases. In contrast, our method do not need such a large memory because the entries of the dictionary in our method are randomly selected from the data that can be stored on disk.

V. Experiments and Results on the UC Merced Land Use Dataset

In this section, we evaluate our proposed method on the UC Merced land use dataset and compare our method with other state-of-the-art methods that have been evaluated with the UC Merced dataset.

The two methods we choose for comparison are spatial pyramid co-occurrence [19] [47] and the unsupervised feature learning method [48]. The spatial pyramid co-occurrence method extends the spatial pyramid kernel, which is a concatenation of the BoW feature vectors of all patches on a multi-resolution grid. The idea is to consider the co-occurrence of a pair of words in each multi-resolution patch where the resulting feature vector contains their concatenation. In contrast, the unsupervised feature learning method follows a conventional procedure of unsupervised feature learning, which comprises two steps, namely dictionary learning and feature coding. The dictionary size is 500. An additional final



(a) (b) (c)

Fig. 14. Classification results on the UCMerced landuse dataset: (a) classification confusion matrix of our proposed method; (b) average accuracy of each class after 20 test runs; (c) legend of the classes.

TABLE I
ACCURACY COMPARISON WITH PREVIOUSLY REPORTED ACCURACIES ON THE UCMERCED DATASET.

Method	BOVW [47]	SPMK [23]	SPCK [47]	SPCK+ [47]	SPCK++ [47]	UFL [48]	Color Histogram [19]	Our Method
Accuracy	71.86%	74.00%	73.14%	76.05%	77.38%	81.67%	81.19%	87.67%

step is feature pooling, i.e., histogram generation. Both the pyramid co-occurrence and the feature learning method have been evaluated on the UC Merced dataset. We follow the same experimental setup for both methods. 80 images from each class of the dataset are randomly selected as training data and the remaining data are used as test data. For our method, we employ the vectorized pixel values from a 3×3 local window as low level feature vectors and use a random dictionary. All classifications are performed in 20 test runs and we present their average accuracy. Then we compared our results with other methods. The corresponding confusion matrix is shown in Fig. 14(a). The average accuracy for each class after 20 test runs is shown in Fig. 14(b). The average accuracy of all classes is 86.42%, as shown in Table. I. The accuracy of our method is 5% better than the best one reported in [48]. In addition, our method is much simpler in terms of both computational effort and memory requirements compared with other methods.

VI. CONCLUSION

In this paper, we focus on remote sensing image classification, including both optical and SAR images. We propose a simple yet quite effective method in the BoW framework. It has two main contributions. The first contribution is that our method does not need to extract any complex low level feature during a pre-processing step, which normally requires a certain amount of computational effort; instead, vectorized pixel values from a very small local window yield a superior BoW performance. The second contribution is that a random dictionary can achieve the same performance as one learned via clustering, which is usually a very time consuming step. In the case of large datasets, this clustering step can make a method infeasible. We performed an extensive investigation of the BoW method and these two contributions have been clearly demonstrated. In addition, we give clear answers to some other relevant but critical questions about BoW feature extraction. These two advantages over conventional methods not only significantly reduce the computational burden but also decrease the memory requirements, thus making the BoW method applicable and scalable to large databases.

REFERENCES

[1] X. Wu, X. Zhu, G. Wu, and W. Ding, "Data Mining with Big Data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97–107, Jan. 2014.

[2] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons," *Int. J. Comput. Vision*, vol. 43, no. 1, pp. 29–44, Jun. 2001.

[3] M. Varma and A. Zisserman, "A Statistical Approach to Texture Classification from Single Images," *Int. J. Comput. Vision*, vol. 62, no. 1-2, pp. 61–81, Apr. 2005.

[4] A. A. Efros and W. T. Freeman, "Image Quilting for Texture Synthesis and Transfer," in *Proc. 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '01*, New York, NY, 2001, pp. 341–346.

[5] M. Varma and A. Zisserman, "Texture classification: are filter banks necessary?," in *Proc. Computer Vision and Pattern Recognition, CVPR*, Madison, WI, 2003, vol. 2, pp. II.692–II.698.

[6] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[8] R. Maree, P. Geurts, J. Piater, and L. Wehenkel, "Random Subwindows for Robust Image Classification," in *Proc Computer Vision and Pattern Recognition, CVPR*, San Diego, CA, 2005, vol. 1, pp. 34–40.

[9] E Nowak, F. Jurie, and B. Triggs, "Sampling Strategies for Bag-of-Features Image Classification," in *Proc. 9th European Conference on Computer Vision ECCV 2006*, Graz, Austria, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 2006, vol. 3954, pp. 490–503.

[10] N. Lazic and P. Aarabi, "Importance of Feature Locations in Bag-of-Words Image Classification," in *Proc. International Conference on Acoustics, Speech and Signal Processing, ICASSP*, Honolulu, HI, 2007, pp. I–641–I–644.

[11] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1265–1278, Aug. 2005.

[12] J. Wu and James M. R., "CENTRIST: A Visual Descriptor for Scene Categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1489–1501, Aug. 2011.

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

[14] G. Zhao, T. Ahonen, J. Matas, and M. Pietikainen, "Rotation-Invariant Image and Video Description With Local Binary Pattern Features," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 1465–1477, Apr. 2012.

[15] J. Sivic and A. Zisserman, “Video Google: A Text Retrieval Approach to Object Matching in Videos,” in *Proc. Ninth IEEE International Conference on Computer Vision, ICCV ’03*, Washington, DC, 2003, vol. 2, pp. 1470–1477.

[16] M. Lienou, H. Maitre, and M. Datcu, “Semantic Annotation of Satellite Images Using Latent Dirichlet Allocation,” *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 1, pp. 28–32, Jan. 2010.

[17] S. Xu, T. Fang, D. Li, and S. Wang, “Object Classification of Aerial Images With Bag-of-Visual Words,” *IEEE Geosci. Remote Sens. Lett.*, vol. 7, no. 2, pp. 366–370, Apr. 2010.

[18] H. Sun, X. Sun, H. Wang, Y. Li, and X. Li, “Automatic Target Detection in High-Resolution Remote Sensing Images Using Spatial Sparse Coding Bag-of-Words Model,” *IEEE Geosci. Remote Sens. Lett.*, vol. 9, no. 1, pp. 109–113, Jan. 2012.

[19] Y. Yang and S. Newsam, “Bag-Of-Visual-Words and Spatial Extensions for Land-Use Classification,” in *Proc. 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS ’10*, New York, NY, 2010, pp. 270–279.

[20] S. Lazebnik and M. Raginsky, “Supervised Learning of Quantizer Codebooks by Information Loss Minimization,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1294–1309, Jul. 2009.

[21] B. Fernando, E. Fromont, D. Muselet, and M. Sebban, “Supervised learning of Gaussian mixture models for visual vocabulary generation,” *Pattern Recognition*, vol. 45, no. 2, pp. 897–907, Feb. 2012.

[22] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek, “Visual Word Ambiguity,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1271–1283, Jul. 2010.

[23] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in *Proc. Conference on Computer Vision and Pattern Recognition, CVPR06*, Washington, DC, 2006, vol. 2, pp. 2169–2178.

[24] J. Yang, K. Yu, Y. Gong, and T. S. Huang, “Linear spatial pyramid matching using sparse coding for image classification,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, FL, 2009, pp. 1794–1801.

[25] Y.-L. Boureau, J. Ponce, and Y. LeCun, “A theoretical analysis of feature pooling in vision algorithms,” in *Proc. International Conference on Machine learning (ICML’10)*, Haifa, Israel, 2010, pp. 111–118.

[26] K. Yu, T. Zhang, and Y. Gong, “Nonlinear Learning using Local Coordinate Coding,” in *Proc. Advances in Neural Information Processing Systems 22 (NIPS 2009)*, Vancouver, Canada, 2009, pp. 2223–2231.

[27] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, “Locality-constrained Linear Coding for Image Classification,” in *Proc. 23rd IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, 2010, pp. 3360–3367.

[28] M. Varma and A. Zisserman, “A Statistical Approach to Material Classification Using Image Patch Exemplars,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 11, pp. 2032–2047, Nov. 2009.

[29] L. Liu and P. Fieguth, “Texture Classification from Random Features,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 3, pp. 574–586, Mar. 2012.

[30] L. Liu, P. Fieguth, Gangyao Kuang, and Hongbin Zha, “Sorted Random Projections for robust texture classification,” in *Proc. International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 2011, pp. 391–398.

[31] L. Liu, P. Fieguth, D. Clausi, and G. Kuang, “Sorted random projections for robust rotation-invariant texture classification,” *Pattern Recognition*, vol. 45, no. 6, pp. 2405–2418, Jun. 2012.

[32] A. Coates and A. Y. Ng, “Learning Feature Representations with K-means,” in *Neural Networks: Tricks of the Trade*, 2012, Springer-Verlag Berlin, Lecture Notes in Computer Science, vol. 7700, pp. 561–580.

[33] A. Coates and A. Y. Ng, “The Importance of Encoding Versus Training with Sparse Coding and Vector Quantization,” in *Proc. 28th International Conference on Machine Learning ICML*, Bellevue, WA, 2011, pp. 921–928.

[34] Y. Huang, Z. Wu, L. Wang, and T. Tan, “Feature Coding in Image Classification: A Comprehensive Study,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 493–506, Mar. 2014.

[35] [Online], “TerraSAR-X Level 1B Product Format Specifications: Issue 1.3, TX-GS-DD-3307,” http://www2.geo-airbusds.com/files/pmedia/public/r460_9_030201_level-1b-product-format-specification_1.3.pdf.

[36] S. Cui, C. O. Dumitru, and M. Datcu, “Semantic annotation in Earth observation based on active learning,” *International Journal of Image and Data Fusion*, vol. 5, no. 2, pp. 152–174, Apr. 2014.

[37] C. C. Chang and C. J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, 2011, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[38] F. Perronnin, J. Sánchez, and T. Mensink, “Improving the Fisher Kernel for Large-Scale Image Classification,” in *Proc. 11th European Conference on Computer Vision ECCV 10*, Hersonissos, Greece, 2010, Lecture Notes in Computer Science, Springer-Verlag, Berlin, vol. 6314, Part IV, pp. 143–156.

[39] B. S. Manjunath and W. Y. Ma, “Texture Features for Browsing and Retrieval of Image Data,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 837–842, Aug. 1996.

[40] J. Singh and M. Datcu, “SAR Image Categorization With Log Cumulants of the Fractional Fourier Transform Coefficients,” *IEEE Trans. Geosci. Remote Sens.*, vol. 51, no. 12, pp. 1–10, Dec. 2013.

[41] R. M. Haralick, K. Shanmugam, and I. Dinstein, “Textural Features for Image Classification,” *IEEE Trans. Syst., Man, Cybern.*, vol. 3, no. 6, pp. 610–621, Nov. 1973.

[42] D. A. Clausi, “An analysis of co-occurrence texture statistics as a function of grey level quantization,” *Can. J. Remote Sensing*, vol. 28, no. 1, pp. 45–62, Feb. 2002.

[43] I. W. Selesnick, R. G. Baraniuk, and N. C. Kingsbury, “The dual-tree complex wavelet transform,” *IEEE Signal Process. Mag.*, vol. 22, no. 6, pp. 123–151, Nov. 2005.

[44] A. Popescu, I. Gavat, and M. Datcu, “Complex SAR image characterization using space variant spectral analysis,” in *Proc. IEEE Radar Conference, RADAR 08*, Rome, Italy, 2008, pp. 1–4.

[45] E. P. Simoncelli and E. H. Adelson, “Non-Separable Extensions of Quadrature Mirror Filters to Multiple Dimensions,” in *Proceedings of the IEEE*, Apr. 1990, vol. 78, pp. 652–664.

[46] J. Singh and M. Datcu, “Mining very high resolution complex-valued SAR images using the fractional Fourier transform,” in *Proc. 9th European Conference on Synthetic Aperture Radar, EUSAR*, Nuernberg, Germany, 2012, pp. 135–138.

[47] Y. Yang and S. Newsam, “Spatial Pyramid Co-occurrence for Image Classification,” in *Proc. 2011 International Conference on Computer Vision ICCV 11*, Washington, DC, 2011, pp. 1465–1472.

[48] A. M. Cheriyadat, “Unsupervised Feature Learning for Aerial Scene Classification,” *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 1, pp. 439–451, Jan. 2014.

Shiyong Cui received the MS degree in photogrammetry and remote sensing from the Chinese Academy of Surveying and Mapping, Beijing, China, in 2009 and the PhD degree in electrical engineering and computer science from the Siegen university, Germany, in 2014.

Since 2009, he has been working with the Remote Sensing Technology Institute (IMF), German Aerospace Center (DLR), Oberpfaffenhofen. His main research interests include computer vision, pattern recognition and machine learning.

Gottfried Schwarz received the Diploma degree in electrical engineering from the Technical University of Mnchen, Mnchen, Germany, in 1976.

Since many years, he has been involved in a number of national and international space projects with the German Aerospace Center (DLR), Munich, Germany. His research interests include the design of deep space instruments from initial engineering studies to detailed design work, modeling of instrument performance, instrument assembly and testing, real time experiment control, instrument check-out and calibration, signal processing, image data compression, feature analysis, classification, data verification and validation as well as data processing and scientific data analysis, in particular of optical and SAR remote sensing data, interpretation of geophysical data with emphasis on retrieval algorithms with inversion techniques, and data mining.

Mihai Datcu (SM'04−F'13) received the M.S. and Ph.D. degrees in electronics and telecommunications from the University Politechnica of Bucharest (UPB), Bucharest, Romania, in 1978 and 1986, respectively.

Since 1981, he has been a Professor in electronics and telecommunications with UPB. Since 1993, he has been a Scientist with the German Aerospace Center (DLR), Munich, Germany. Currently, he is a Senior Scientist and Image Analysis Research Group Leader with the Remote Sensing Technology Institute (IMF), DLR, Coordinator of the CNES-DLR-ENST Competence Centre on Information Extraction and Image Understanding for Earth Observation, and a Professor with Paris Institute of Technology/GET Telecom Paris. From 1991 to 1992, he was a Visiting Professor with the Department of Mathematics, University of Oviedo, Oviedo, Spain, and from 2000 to 2002 with the Universitouis Pasteur, and the International Space University, both in Strasbourg, France. From 1992 to 2002, he was a Longer Invited Professor with the Swiss Federal Institute of Technology ETH Zrich, Zrich, Switzerland. In 1994, he was a Guest Scientist with the Swiss Center for Scientific Computing (CSCS), Manno, Switzerland, and in 2003, he was a Visiting Professor with the University of Siegen, Siegen, Germany. His research interests include Bayesian inference, information and complexity theory, stochastic processes, model-based scene understanding, image information mining, for applications in information retrieval and understanding of high-resolution SAR and optical observations.

Dr. Datcu is a member of the European Image Information Mining Coordination Group (IIMCG). In 1999, he received the title Habilitation diriger des recherches from Universitouis Pasteur, Strasbourg, France.