Hindawi
Journal of Sensors
Volume 2018, Article ID 6257810, 14 pages
https://doi.org/10.1155/2018/6257810

Research Article

Tile-Based Semisupervised Classification of Large-Scale VHR Remote Sensing Images

Haikel Alhichri, Essam Othman, Mansour Zuair, Nassim Ammour, and Yakoub Bazi

Computer Engineering Department, College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia

Correspondence should be addressed to Haikel Alhichri; haikel.alisr@gmail.com

Received 14 September 2017; Revised 14 January 2018; Accepted 21 January 2018; Published 5 April 2018

Academic Editor: Biswajeet Pradhan

This paper deals with the problem of the classification of large-scale very high-resolution (VHR) remote sensing (RS) images in a semisupervised scenario, where we have a limited training set (less than ten training samples per class). Typical pixel-based classification methods are unfeasible for large-scale VHR images. Thus, as a practical and efficient solution, we propose to subdivide the large image into a grid of tiles and then classify the tiles instead of classifying pixels. Our proposed method uses the power of a pretrained convolutional neural network (CNN) to first extract descriptive features from each tile. Next, a neural network classifier (composed of 2 fully connected layers) is trained in a semisupervised fashion and used to classify all remaining tiles in the image. This basically presents a coarse classification of the image, which is sufficient for many RS application. The second contribution deals with the employment of the semisupervised learning to improve the classification accuracy. We present a novel semisupervised approach which exploits both the spectral and spatial relationships embedded in the remaining unlabelled tiles. In particular, we embed a spectral graph Laplacian in the hidden layer of the neural network. In addition, we apply regularization of the output labels using a spatial graph Laplacian and the random Walker algorithm. Experimental results obtained by testing the method on two large-scale images acquired by the IKONOS2 sensor reveal promising capabilities of this method in terms of classification accuracy even with less than ten training samples per class.

1. Introduction

The intent of the classification process is to categorize all pixels in a remote sensing (RS) image into one of several land-cover classes. Most classification solutions have used RS images with small sizes (typically less than 1000×1000 pixels). However, the latest generation of satellite-based imaging sensors (Pleiades, Sentinel, etc.) acquires big volumes of Earth's images with high spatial, spectral, and temporal resolution. This leads to images of very large sizes and creates new challenges for land-use classification algorithms. In this case, traditional pixel-based algorithms become unfeasible due to (1) increased computational and memory costs, (2) increased spectral variation within the same land-cover class, and (3) increased variability in the data distributions over this large-scale image, and (4) the need to collect a large set of training samples to properly model the underlying distribution of the data in the image. The latter issue may be caused by the increased resolution which heightens the effect of lighting conditions, the depth dimension, and the sensor's noise. The same land-covers and even the same objects can be found in images belonging to different classes. Thus, there is a great need to develop efficient solutions to classify large-scale images.

One solution to this problem is to use object-based techniques. This can be accomplished first via segmenting the large image into regions and then performing object analysis on the regions. However, segmenting large images is, in itself, still a difficult problem [1]. Another solution is to divide the image into tiles and then perform coarse classification of the image by classifying the tiles instead of pixels as was introduced in [2–4].

The problem of tile classification becomes then similar to object classification or recognition in computer vision applications. In those applications, many local descriptors, like bag of words (BOW) [5], local binary patterns (LBP) [6], and histograms of oriented gradients (HOG) [7], with their invariance to geometric and photometric transformations, have been proven effective especially for object recognition. They can be extracted both in sparse and in dense ways. Recently, deep convolutional neural networks (CNN) [8–10] have shown outstanding results in many computer vision applications such as image classification [11], object recognition [12], face recognition [13], medical image analysis [14], speech recognition [15], and traffic flow prediction [16]. Several CNN architectures have been already proposed and tested in computer vision tasks, and most of them have been implemented and made available online [17].

CNNs use images directly as input, and they automatically learn a hierarchy of features instead of using handcrafted features. This is accomplished by successively convolving the input image with learned filters to build up a hierarchy of feature maps. The hierarchical approach allows learning more and more complex and highly descriptive set of features. One disadvantages of CNN is the high computational costs during the training phase due to the high number of weights to be estimated (in the range of hundreds of thousands). Another stringent requirement is the need for a huge number of training images [18]. One way to solve this problem is to use CNNs that have been pretrained on huge auxiliary datasets and then transfer the knowledge embedded in them to help extract highly descriptive features in our new domain. These highly descriptive features are then used to train another classifier (such as a smaller neural network) for our particular recognition task.

In the remote sensing community, some works have introduced CNN as a solution to classification and object detection problems [19–23]. For the scene classification problem, the reader is referred to this work by Cheng et al. [24] for a recent review of the state-of-the-art review in the area. As a set of sample work, Castelluccio et al. proposed a solution for the classification of RS scenes by starting with a pretrained CNN and then fine-tuning it on the target data in order to improve the accuracy [19]. Zhang et al. proposed a method called gradient boosting random convolutional network (GBRCN), which trains and fuses the result of several CNNs for the purpose of RS scene classification [22]. Chen et al. proposed a method to detect small objects such as vehicles in satellite images using a CNN to extract and learn rich features from the training data [21]. Marmanis et al. extracted feature representations using a pretrained CNN and then fed them into another CNN, which is trained in a supervised manner [20]. Another more recent work proposes a novel new feature for scene classification called bag of convolutional features (BoCF) [23]. This is similar to bag of words concept, except that the codebook is composed of convolutional feature vectors collected from the 5th layer of the VGGNet-16 CNN when it is applied to all training and testing scenes.

Another important issue in RS classification is that the collection of a statistically significant and representative amount of training (labelled) samples is a tedious task. Thus, the RS community has introduced semisupervised learning (SSL) methods to tackle this problem [25–29]. SSL methods attempt to exploit the wealth of information provided by unlabelled data, besides the few available labelled data, to improve the performance of the classifier. SSL is based on the assumption that points within the same structure (such as a cluster or a manifold) are likely to have the same label [29]. We can use the large amount of unlabelled data to uncover such structure. For example, in RS images, it is reasonable to assume that samples are likely to have similar labels, if they have close spectral information or if they are neighbours spatially. Many machine-learning researchers have found that unlabelled data, when used in conjunction with a small amount of labelled data, can produce considerable improvement in learning accuracy [30].

Semisupervised learning is natural for deep CNN architectures since the latter requires huge training data and in practice, either we never have enough labelled data or it is quite costly to get them. Nonlinear embedding algorithms are popular for use with shallow semisupervised learning techniques such as kernel methods, but they can be applied to deep multilayer architectures as well, as a regularizer either at the output layer or on the hidden layers of the architecture [31]. Graph-based techniques are one way to perform semisupervised learning. In this approach, one tries to preserve the structure present in the unlabelled data by building a graph where each sample data point is a node and the edges encode the similarity between data points. In the literature, graph Laplacian is embedded at the hidden layer and optimized in a joint way within the error function of the network [29]. Another approach is to use the graph Laplacian as a regularizer at the output layer such as in Markov random field (MRF).

In this work, we present an effective solution for the classification of large-scale VHR RS images, where the number of labelled samples is limited (no more than ten tiles per class). To deal with the huge size of the VHR images, we subdivide the large image into square tiles and classify them instead of classifying pixels. This is our first contribution in this paper. Our second contribution is presenting a deep architecture for semisupervised classification of large-scale images. This deep architecture is composed of a pretrained CNN, for feature extraction, followed by two fully connected layers for classification. The proposed solution accomplishes semisupervised learning via simultaneous embedding of a graph Laplacian in the hidden layer followed by a spatial graph Laplacian after the output layer for regularization. The experimental results, carried out on two large-scale VHR images, have shown that embedding a graph Laplacian at the feature level and at the output level simultaneously provides significant improvement in the classification result.

The rest of the paper is organized as follows: in Section 2, we present the details of proposed method. In Section 3, we present the experimental results and discuss them. Finally, we present our concluding remarks and future works in Section 4.

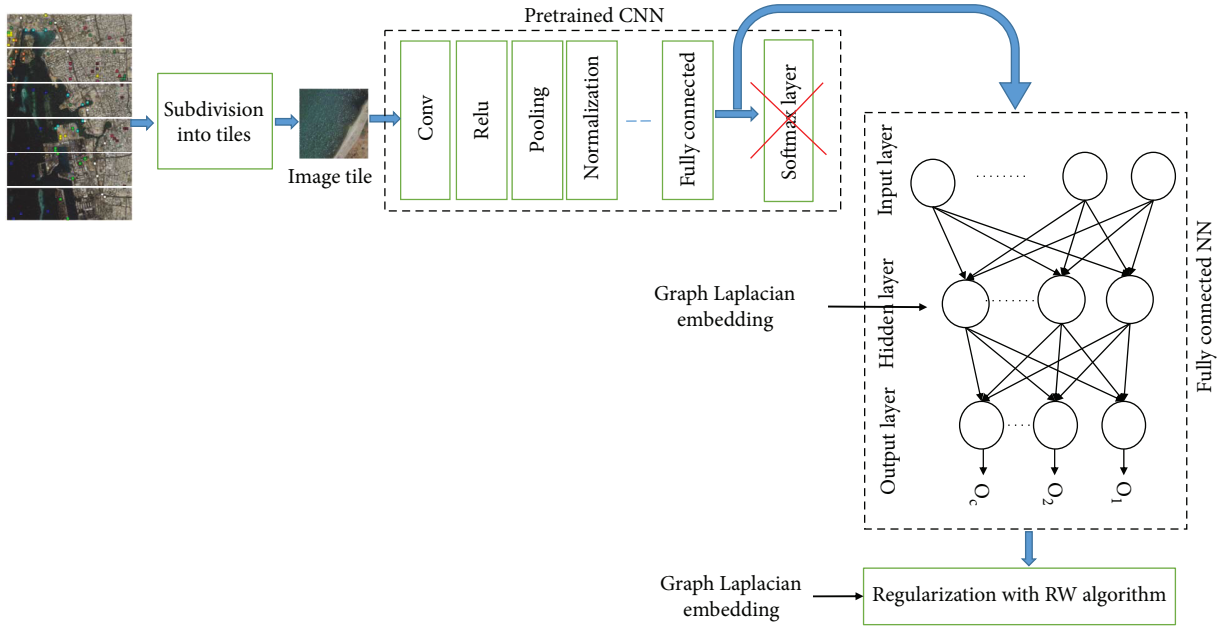

Figure 1: Overview of the proposed deep CNN solution for the semisupervised classification of large RS images.

2. Materials and Methods

The general overview of our proposed solution is shown in Figure 1. The first step in this solution is to divide the large image into a grid of tiles, and the goal is to classify these tiles instead of pixels to produce a coarse tile-based classification of the RS image. Let I be a large VHR RS image that is divided into a grid of $\mathbf{N}_t$ tiles $\{\mathbf{I}_i\}_{i=1}^{\mathbf{N}_t}$, where $\mathbf{I}_i \in \mathcal{R}^{w \times w}$ represents tile $\mathbf{i}$ of size $w \times w$. The second step in the algorithm is feature extraction using one of the pretrained CNNs found in the literature. Let $\{\mathbf{x}_i\}_{i=1}^{\mathbf{N}_t}$ be the set of features extracted from each one of the $\mathbf{N}_t$ tiles. Furthermore, let $\mathrm{Tr} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$ be a training set of labelled sample tiles, where $y_i \in \{1, 2, \ldots, C\}$ are the corresponding class labels.

The classification step is carried out using a fully connected neural network (NN), as shown in Figure 1, with one hidden layer and an output (softmax) layer. As mentioned previously, using this proposed architecture, we can implement semisupervised learning in different ways. We can embed the graph Laplacian at the hidden layer of the fully connected neural network (i.e., at the feature level). We can also embed a graph Laplacian at the output level, which implements what is called output regularization. Finally, we can implement a suitable combination thereof. Our final proposed solution shown in Figure 1 embeds a graph Laplacian (spectral or spatial) in the hidden layer of the NN and also adds an additional layer that performs regularization of the output labels. In the remainder of this section, we explain how to use a pretrained CNN for feature extraction. Then the next subsection shows how to perform classification using a fully connected neural network. Finally, we explain how to achieve semisupervised learning via embedding a graph Laplacian at the hidden layer of the neural network and also embedding another graph Laplacian to perform output regularization spatially.

2.1. Feature Extraction from Tiles. The second step in the proposed solution is to extract feature descriptors from the image tiles. Several types of handcrafted feature descriptors are presented in the literature for image representation, such as bag of words (BOW) [5], local binary patterns (LBP) [6], or histograms of oriented gradients (HOG) [7]. However, more recently, the new state-of-the-art approach is to use feature descriptors that can be learned instead of handcrafted features. CNNs play a major role in this regard as they have been proven effective in feature learning from images. However, as CNN needs a huge amount of training data, one can resort to pretrained CNN as a compromise solution. In this case, a pretrained CNN becomes like a feature extraction tool instead of a classifier as it is originally designed. This approach exploits the huge information embedded in pretrained CNNs to provide highly descriptive features from the tiles.

A CNN is composed of several layers that do different functions such as convolution, pooling, activation function, and classification as shown in Figure 1. The last layer is always a softmax layer which is used for classification. Given this deep nature, one can extract features at different layers of the CNN. In this work, we take the output of the hidden layer before the softmax layer as feature vector representation. Thus, given a CNN with L layers, we feed each tile image $\mathbf{I}_i$, $i = 1, \ldots, \mathbf{N}_t$ as input to the CNN and generate a feature representation $\mathbf{x}_i \in \mathcal{R}^D$ at layer k, of the CNN, with $k = L - 1$, that is,

$$\mathbf{x}_i = f_k^{\mathrm{CNN}}\left(\ldots f_2^{\mathrm{CNN}}\left(f_1^{\mathrm{CNN}}(\mathbf{I}_i)\right)\right), \quad i = 1, \ldots, \mathbf{N}_t, \tag{1}$$

where D is the dimensionality of the feature vector.

2.2. Classification via a Fully Connected Layer. Given a training set $\mathrm{Tr} = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$ composed of feature vectors and their corresponding labels, the hidden layer takes the

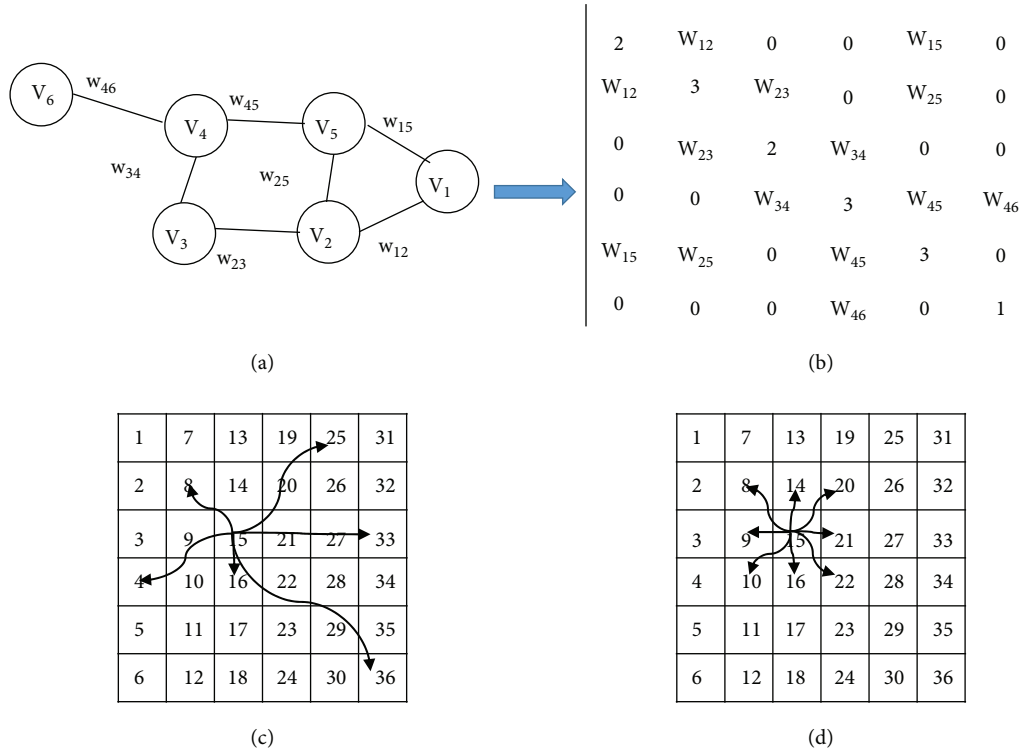

Figure 2: Illustration of graph Laplacian building. (a) Sample graph with 6 vertices and 7 edges. (b) Laplacian matrix. (c) Spectral graph Laplacian. Each tile (a node in the graph) is connected to the k-most similar tiles. (d) Spatial graph Laplacian, each tile (a node in the graph) is connected to its 8 neighbours.

input x_i and maps it to another representation $h_i^{(1)} \in \Re^{d^1}$ of dimension d^1 through the nonlinear activation function f as follows:

$$h_i^{(1)} = f\left(\mathbf{W}^{(1)} x_i\right), \tag{2}$$

where $\mathbf{W}^{(1)} \in \Re^{d^1 \times d}$ is the mapping weight matrix. A typical choice of the activation function is the sigmoid function, that is, $f(v) = 1/(1 + \exp(-v))$. For simplicity, we omit the bias vector in the expression as it can be incorporated as an additional column vector in the mapping matrix; then in that case, the feature vector should be appended by the value 1.

The *softmax* layer performs multiclass classification and takes as input the resulting hidden representation $h_i^{(1)}$. It produces, as a result, an estimate of the posterior probability for each class label $c = 1, 2, \ldots, C$ as follows:

$$p(\widehat{y}_i = c \mid x_i) = \frac{\exp\left(\left(w_c^{(2)}\right)^T h_i^{(1)}\right)}{\sum_{j=1}^{C} \exp\left(\left(w_j^{(2)}\right)^T h_i^{(1)}\right)}, \tag{3}$$

where $\mathbf{W}^{(2)} = [w_1^{(2)}\, w_2^{(2)} \ldots w_C^{(2)}] \in \Re^{d \times C}$ are the weights of the *softmax* layer and the superscript $(\bullet)^T$ refers to the transpose operation. To learn the weights $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}\}$ representing the complete network structure, we minimize the training error on the labelled training data. The cost function is then formulated as follows:

$$J\left(\theta, D^{(1)}\right) = \arg\ \min E_{\text{net}}\left(D^{(1)}\right), \tag{4}$$

where $D^{(l)}$ is the set of labelled data and $E_{\text{net}}(D^{(l)})$ is the cross-entropy loss, which measures the error between the actual network outputs and the desired outputs of the labelled source data. As the outputs of the network are probabilistic, we propose to maximize the log-posterior probability to learn the network weights, which is equivalent to minimizing the so-called cross-entropy error:

$$E_{\text{net}}\left(D^{(l)}\right) = -\frac{1}{n} \sum_{i=1}^{n_s} \sum_{k=1}^{K} 1(y_i = k) \ln\left(\frac{\exp\left(\left(w_k^{(2)}\right)^T h_k^{(1)}\right)}{\sum_{j=1}^{K} \exp\left(\left(w_j^{(2)}\right)^T h_l^{(1)}\right)}\right), \tag{5}$$

where $1(\bullet)$ is an indicator function that takes 1 if the statement is true otherwise it takes 0 and the superscript T refers to matrix transpose.

2.3. Graph-Based Semisupervised Learning. The idea of graph-based semisupervised learning is to build a graph that connects similar sample data points to each other, and then use this graph to propagate estimated labels among similar data points. This follows directly from the assumption that similar data points should have similar labels, which the graph tries to encode.

Figure 2 illustrates how a Laplacian matrix is built using a simple example. Let $G = (V, E)$ be a graph with vertices $\mathbf{v} \in V$

and edges $e \in E$. Let e_{ij} denote an edge spanning two vertices $\mathbf{v}_i$ and $\mathbf{v}_j$ and ω_{ij} is the associated weight. The degree of a vertex $\mathbf{v}_i$ is $d_i = \sum \omega_{ij}$ for all edges e_{ij} incident on $\mathbf{v}_i$. Then the tile-based Laplacian matrix indexed by vertices $\mathbf{v}_i$ and $\mathbf{v}_j$ is given by:

$$\mathbf{L}_{ij} = \begin{cases} d_{ij}, & \text{if } i = j, \\ -\omega_{ij}, & \text{if } i \text{ and } j \text{ are connected}, \\ 0, & \text{otherwise}. \end{cases} \quad (6)$$

In our case, each tile image is associated with a vertex in the graph, while edges are defined in several ways. The k-nearest graph, for example, connects each vertex to the k-most similar vertices. The number k is usually left as a parameter to be determined. Another approach is to connect each vertex to all vertices whose similarity is below a certain threshold value. This is known as a spectral graph and is illustrated in Figure 2(d).

However, here, the tiles are also spatially dependent. Thus, we can also consider adjacency as a similarity, that is, we can make the assumption that neighbouring tiles are similar and should have similar labels. In this case, the number of connections per node is usually limited to two choices: 8 or 4 depending on whether we want to consider all 8 neighbouring tiles or only 4. This is known as a spatial graph and is illustrated in Figure 2(c).

Finally, a weight w_{ij} is associated with each edge e_{ij} in the graph. This weight is defined as the similarity between image tiles corresponding to vertices $\mathbf{v}_i$ and $\mathbf{v}_j$. It can be calculated using many types of similarity measures or distance measures. A common choice for obtaining these weights is the Gaussian weighting function, that is, $\omega_{jk} = \exp(-\beta \|\mathbf{x}_j - \mathbf{x}_k\|^2)$ and β is a free parameter (usually set to 1).

2.3.1. Embedding Graph Laplacian in the Hidden Layer. The first approach to implement semisupervised learning is by embedding a spectral or spatial graph Laplacian in the hidden layer. The fully connected neural network is supplied with labelled and unlabelled data. Then to learn the weights $\theta = \{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}\}$ representing the complete network structure, we propose to simultaneously minimize: (i) the training error on the labelled data and (ii) the energy of the graph Laplacian built on all unlabelled tiles. The proposed cost function is then formulated as follows:

$$J\left(\theta, D^{(1)}, D^{(\mathrm{u})}\right) = \arg\min E_{\mathrm{net}}\left(D^{(l)}\right) + \lambda_1 \mathrm{Lap}\left(D^{(u)}\right), \quad (7)$$

where λ_1 is a regularization parameter. The first term is the same as (4), while the second term is added for graph regularization. This basically minimizes the energy of the graph and can be written as follows:

$$\mathrm{Lap}\left(D^{(u)}\right) = \frac{1}{2} \sum_{e_{jk} \in E} \omega_{jk} \left\| \mathbf{W}^{(1)} \mathbf{x}_j - \mathbf{W}^{(1)} \mathbf{x}_k \right\|_2^2. \quad (8)$$

It can be shown that (8) can be written in the following matrix form:

$$\mathrm{Lap}\left(D^{(u)}\right) = \frac{1}{2} \mathrm{Trace}\left(\left(\mathbf{W}^{(1)} \mathbf{X}^{(u)}\right) \mathcal{L} \left(\mathbf{W}^{(1)} \mathbf{X}^{(u)}\right)^{\mathrm{T}} \right), \quad (9)$$

where $\mathbf{X}^{(u)} = [\mathbf{x}_1 \ \cdots \ \mathbf{x}_{n_u}] \in \mathcal{R}^{n_u \times d^1}$ represents the unlabelled samples.

By substituting the terms in (5) and (9), the total cost function $J(\theta)$ is then given by

$$\begin{aligned} J\left(\theta, D^{(1)}, D^{(u)}\right) = \arg\min & - \frac{1}{n} \sum_{i=1}^{n_l} \sum_{k=1}^{K} 1(y_i = k) \\ & \cdot \ln \left(\frac{\exp\left(w_k^{(2)^{\mathrm{T}}} h_k^{(1)} \right)}{\sum_{l=1}^{K} \exp\left(w_l^{(2)^T} h_l^{(1)} \right)} \right) + \frac{\lambda_1}{2} \\ & \cdot \mathrm{Trace}\left(\left(\mathbf{W}^{(1)} \mathbf{X}^{(u)}\right) \mathcal{L} \left(\mathbf{W}^{(1)} \mathbf{X}^{(u)}\right)^{T} \right). \end{aligned} \quad (10)$$

2.3.2. Spatial Regularization of the Output Layer. Another approach to exploit the graph Laplacian is an output regularizer based on the random Walker (RW) algorithm. This technique is similar to a Markov random field (MRF) approach; however, unlike the MRF, it yields a closed-form solution as opposed to an iterative one [32]. This regularization step needs a spatial graph Laplacian and posterior probability estimates as initial inputs. Then the RW algorithm can improve the classification result by minimizing an energy function of the graph Laplacian, defined as follows:

$$\min_f \sum_{e_{ij} \in E} \omega_{ij} \left\| \mathbf{f}_i - \mathbf{f}_j \right\|^2 + \lambda_2 \sum_{\mathbf{v}_i \in \mathbf{V}} \mu_i \| \mathbf{f}_i - \mathbf{f}_i^* \|^2, \quad (11)$$

where $\mathbf{f}_i$ and $\mathbf{f}_j$ are now probability vectors (of dimension C) associated with vertices (tiles) $\mathbf{v}_i$ and $\mathbf{v}_j$, respectively. The first term is related to image smoothness since it computes the norm between pixels. The second term is related to data fidelity since it computes a norm between the initial estimation $\mathbf{f}_i^*$ (provided by the fully connected NN) and the new label estimates $\mathbf{f}_i$. Finally, μ_i and λ_2 are local and global weights, respectively, enforcing that fidelity. Moreover, the local weights μ_i could be set equal to d_i, but for simplicity, we suppose $\mu_i = 1$ for all tiles in the image.

The set of vertices of the graph can be written as $\mathbf{V} = \mathbf{V}_T \cup \mathbf{V}_U$, where $\mathbf{V}_T$ and $\mathbf{V}_U$ are the set of labelled and unlabelled vertices, respectively. In this case, the Laplacian matrix $\mathbf{L}$ can be written as $\mathbf{L} = \begin{bmatrix} \mathbf{L}_T & \boldsymbol{B} \\ \boldsymbol{B} & \mathbf{L}_U \end{bmatrix}$. After differentiating, setting the result to 0 and solving the matrix

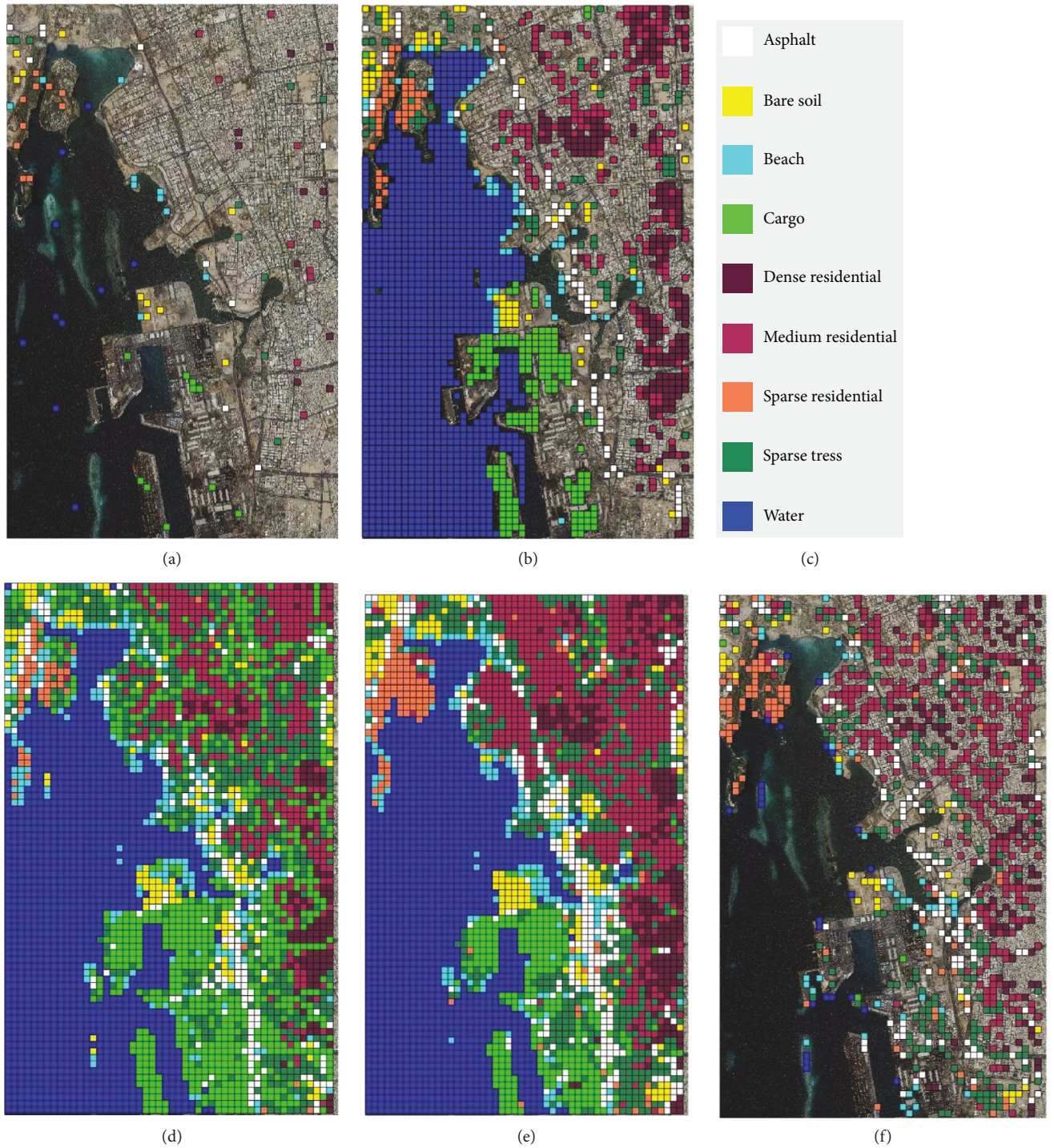

FIGURE 3: Large VHR dataset used and its qualitative results. (a) Original image of size 6500×10400 with example training tiles (10 per class), (b) testing map (ground truth), (c) legend, (d) NN classification result (OA = 89%), (e) final map (OA = 92.2%), and (f) tiles corrected by semisupervised learning.

equation, we found that the probabilities associated with the unlabelled tiles are obtained by the following closed-form solution:

$$\mathbf{F}_U = (\mathbf{L}_U + \lambda_2 \mathbf{I})^{-1} \left(-\mathbf{B}^T \mathbf{F}_T + \lambda \mathbf{F}_U^*\right), \tag{12}$$

where

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_T \\ \mathbf{F}_U \end{bmatrix} \tag{13}$$

and $\mathbf{F}_U^*$ are the initial estimated probabilities (posterior probabilities of the fully connected NN classifier) associated with the unlabelled tiles. Note that $\mathbf{L}_U + \lambda_2 \mathbf{I}$ is a positive semidefinite matrix, and $\mathbf{L}$ is a sparse (Laplacian) matrix.

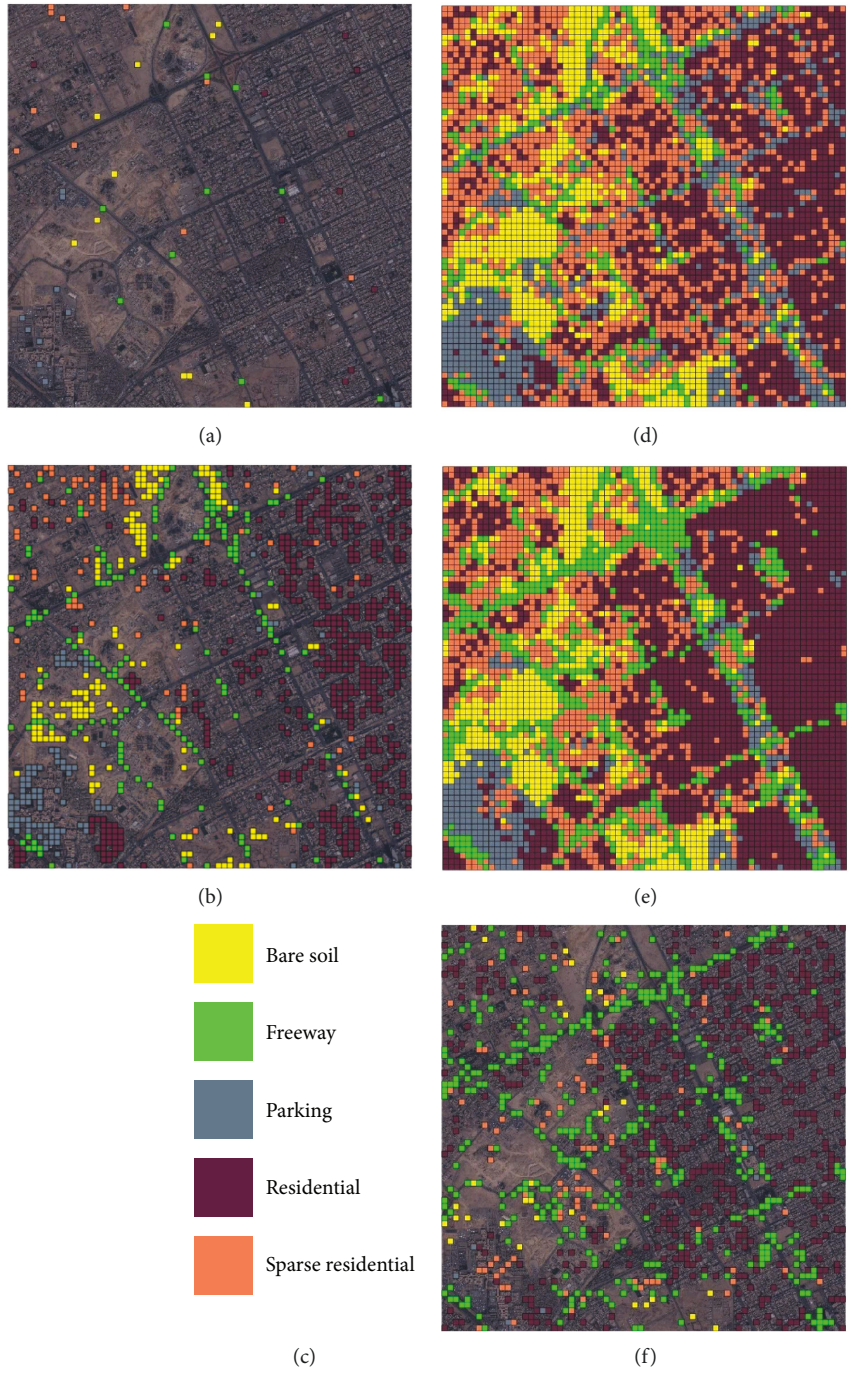

Figure 4: Large VHR dataset used and its qualitative results. (a) Original image of size 13440×13440 with example training tiles (10 per class), (b) testing map (ground truth), (c) legend, (d) NN classification result (OA = 91%), (e) final map (OA = 96.5%), and (f) tiles corrected by semisupervised learning.

Thus, the linear system of equations in (12) can be solved in linear time.

3. Results

3.1. Dataset Description. We tested the proposed solution on two large-scale VHR RS images. The first image is of size 6500×10400 taken over the city of Jeddah in Saudi Arabia by the IKONOS-2 sensor in July 2004 (see Figure 3(a)). The image has three spectral bands with a spatial resolution of 1 m. It has been divided into 3977 tiles of size 128×128, of which 1949 tiles are unlabelled, while 2028 are labelled into nine land-cover types including asphalt (73 tiles), bare soil (61 tiles), beach (60 tiles), cargo (174 tiles), dense residential (203 tiles), medium residential (193 tiles), sparse residential (47 tiles), trees (74 tiles), and water (1216 tiles).

The second image is of size 14000×14000 and is taken from the city of Riyadh by GeoEye sensor in June 2010

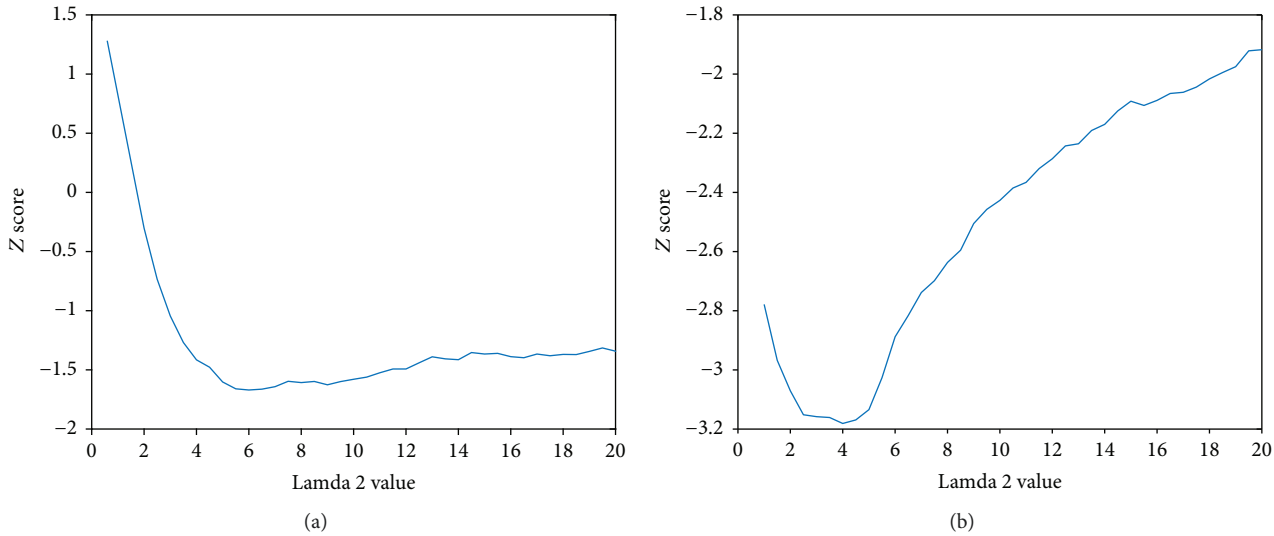

Figure 5: Results of output regularization via spatial graph Laplacian with sensitivity analysis of the parameter λ_2 in (11), (a) for Riyadh dataset and (b) for the Jeddah dataset.

(see Figure 4(a)). The image has three spectral bands with a spatial resolution of 0.5 m. It has been divided into 4900 tiles of which 3860 are unlabelled, while 1040 are labelled into five land-cover types including bare soil (163 tiles), freeway (185 tiles), parking (109 tiles), residential (486 tiles), and sparse residential (97 tiles).

3.2. Experimental Setup. Both large images have been subdivided into a grid of equal size tiles. The Jeddah image is divided in tiles of size 128×128 pixels, while the Riyadh image tiles are of size 196×196 pixels. We start the division from the top left corner, and any leftover pixels (in case the size of the large image is not a multiple of 128) appear at the right and bottom sides of the image. The ground truth classification maps are created manually by inspecting a random set of tiles (the test set) one by one and assigning one of the 9 labels to all tiles in the test set. Testing maps for both the Jeddah and Riyadh images are shown in Figures 5(b) and 6(b), respectively. For both images, we extract feature representations from all tiles including HOG, LBP, BOW, and several pretrained CNN architectures. The set of pretrained CNNs used includes the imagenet-vgg-very-deep-16 and imagenet-vgg-m models by the VGG group at the University of Oxford [33, 34], the GoogleNet model developed by a team from Google Inc. [35], and Microsoft's ResNet-152 model [36]. These CNNs are selected because they have won the first and second places in the ImageNet Large-Scale Visual Recognition Challenge-(ILSVRC-) 2014 and 2015 challenges. Table 1 shows the sizes of each feature representation.

As for the assessment method, we use the Overall Accuracy (OA) and Average Accuracy (AA) measures. To assess the statistical significance of the improvements achieved by applying the semisupervised learning, we computed McNemar's test, which is based on the standardized normal test statistic:

$$Z_{ij} = \frac{f_{ij} - f_{ji}}{\sqrt{f_{ij} - f_{ji}}}, \tag{14}$$

where Z_{ij} measures the pairwise statistical significance of the difference between the accuracies of the ith and jth classifiers. The variable f_{ij} stands for the number of samples classified correctly and wrongly by the ith and jth classifier, respectively. Thus, f_{ij} and f_{ji} are the counts of the classified samples on which the two classifiers disagree. At the commonly used 5% level of significance, the improvement of classifier j over i is said to be significant if $Z_{ij} < -1.96$. Finally, all experiments are repeated 10 times with different random training tiles and then the average accuracy values are presented.

3.3. Comparison of Various Pretrained CNN and Finding the Best Hidden Layer Size. In this first experiment, we compare the performance of feature representations obtained by the pretrained CNNs and the ones obtained by BOW, LBP, and HOG. We also investigate in this experiment the sensitivity of the results with respect to the hidden layer size for all the types of feature descriptors considered in this study. We train a fully connected NN with the following hidden layer sizes 32, 64, 128, 256, 512, and 1024. We perform the experiment 10 times; each time, we select a different set of training sample tiles per class randomly and average the accuracy results.

Figure 7 shows the overall accuracies (OA) achieved for each type of feature descriptor and with different hidden layer sizes. It is clear from this experiment that CNN features are quite superior to all other types of handcrafted features regardless of the hidden layer size. This confirms other

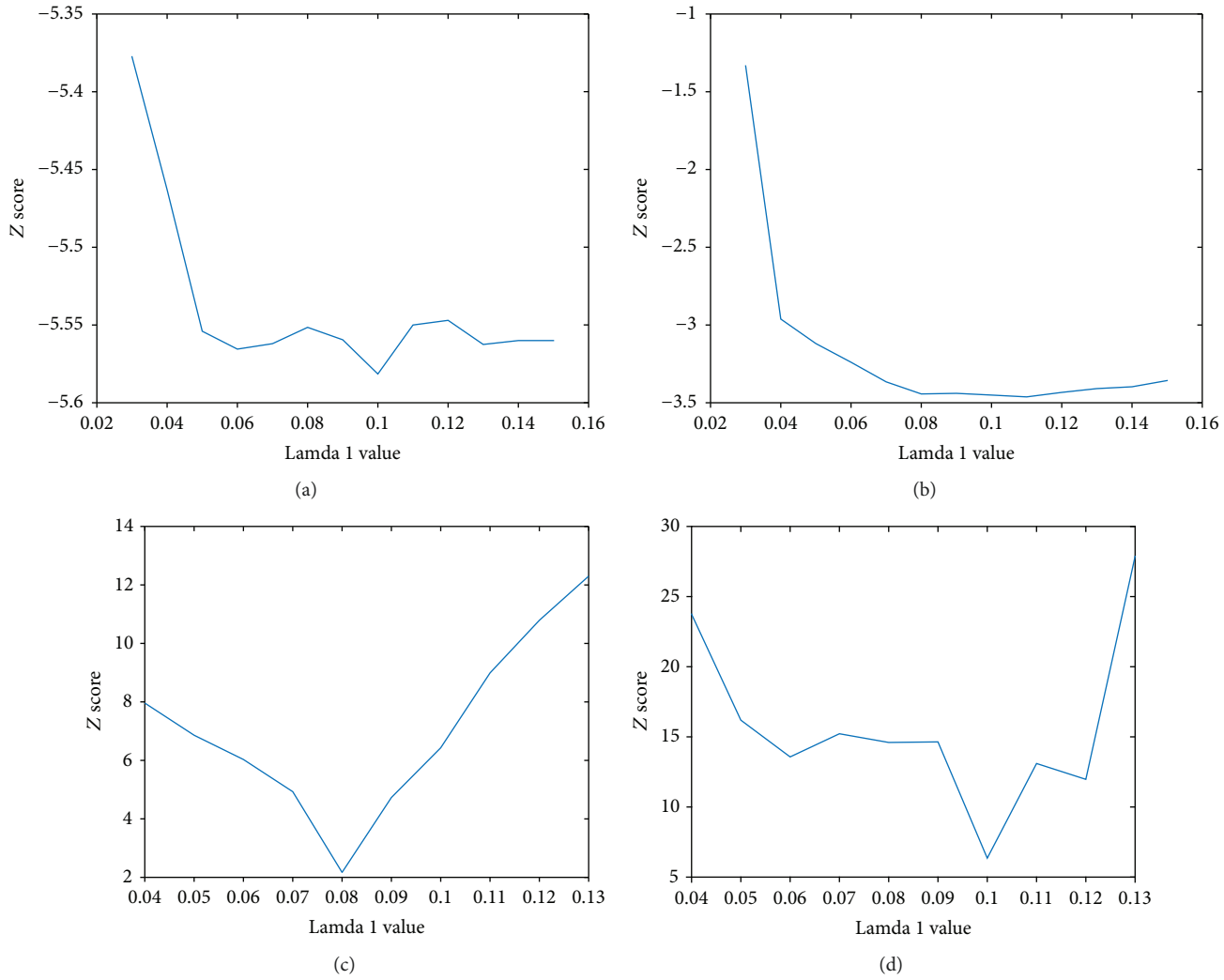

Figure 6: Results of embedding a graph Laplacian in the hidden layer with sensitivity analysis of the parameter λ_1 in (10). (a) Riyadh dataset with spectral graph. (b) Jeddah dataset with spectral graph. (c) Riyadh dataset with spatial graph. (d) Jeddah dataset with spatial graph.

Table 1: Parameters used by various feature extraction techniques.

Feature type	Feature vector length	Note
HOG	4365	
LBP	256	8 neighbourhood, uniform pattern
BOW	4000	Vocabulary size = 4000
CNN—vgg-very-deep-16	4096	
CNN—GoogLeNet	1024	
CNN—vgg-m	1024	
CNN—ResNet-152	2048	

results about CNN that is reported in the object recognition literature.

Furthermore, if we consider CNN type features only, VGG very-deep-16 pretrained CNN, in particular, provided the best OA. We can also conclude that in general the range [64, 256] constitutes the best values for the hidden layer size. For the remainder of this paper, the hidden layer size of 256 and the VGG very-deep-16 pretrained CNN will be used in all experiments.

3.4. Semisupervised Learning Results. In this set of experiments, we present the results of applying semisupervised learning via graph Laplacian. We first implement embedding a spatial graph Laplacian as an output regularizer based on the RW algorithm without embedding any graph in the hidden layer. This step has a regularization parameter λ_2 as shown in (11). In Figure 5, we show the Z score achieved for different values of the regularization parameter in the range of $[1 \ldots 20]$. From these results, we learn that output regularization does not provide significant improvement for the Riyadh dataset as the Z score is greater than -1.96.

However, the improvement for the Jeddah dataset is significant (<-1.96). We also learn that the best value for the parameter λ_2 is around 4 to 6; thus, we decided to fix this parameter to 5 for optimal performance of the method.

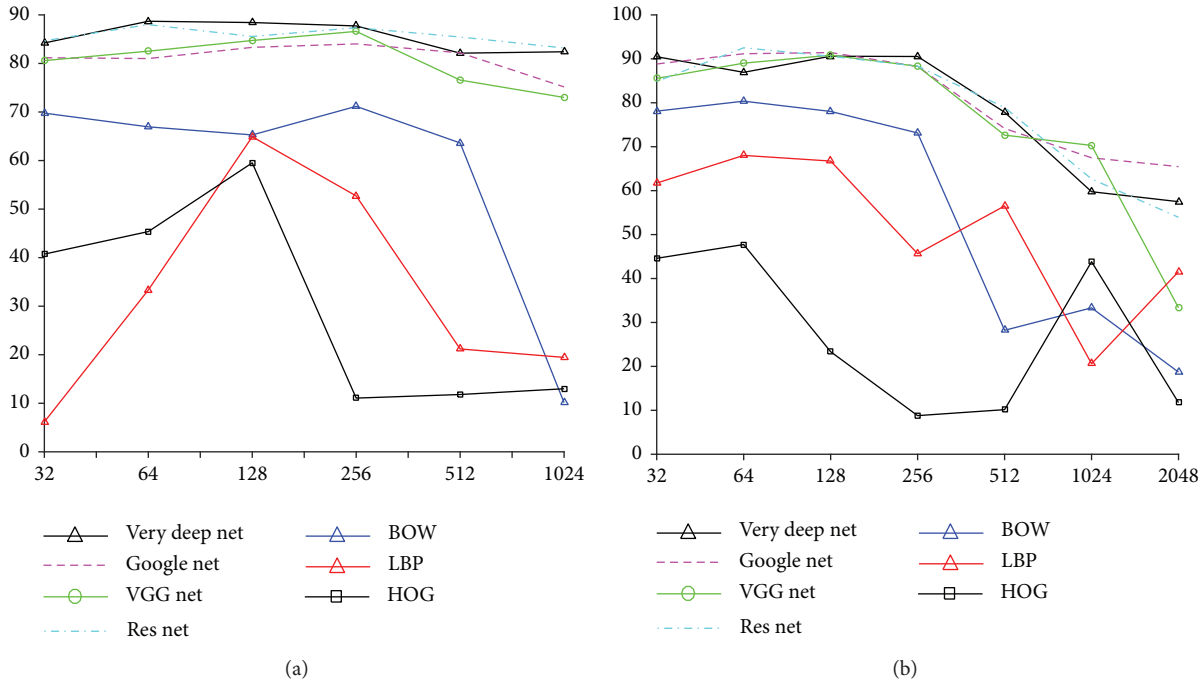

Figure 7: Comparison between various feature descriptors in terms of overall accuracy (OA). (a) OA for the Jeddah-IKONOS2 dataset and (b) OA for the Riyadh-GEOEYE dataset.

In the second experiment, we present the results of embedding a graph Laplacian in the hidden layer. We investigated embedding both a spectral graph Laplacian and spatial graph Laplacian. The cost function to be optimized, shown in (10), has a regularization parameter λ_1. Figure 6 shows the Z score achieved for different values of the regularization parameter in the range of $[0.04 \ldots 0.15]$. From these results, we learn that embedding a spatial graph Laplacian in the hidden layer degrades the classification accuracy significantly as the Z scores in Figures 6(c) and 6(d) are all above -1.96. On the other hand, embedding a spectral graph Laplacian in the hidden layer does provide significant improvement for both datasets as the Z score is less than -1.96. As for the best value for the parameter λ_1, it is clear, from Figures 6(a) and 6(b), that it is around the values 0.08 to 0.1; thus, we decided to fix this parameter to 0.1 for optimal performance of the method.

Next, we present the results of the complete system shown in Figure 1. The results for the Jeddah and Riyadh image are shown in Tables 2 and 3, respectively. These tables show the OA, AA, and per class accuracy for 5 different scenarios: (1) using the fully connected NN only, (2) using the NN followed by spatial regularization using RW algorithm, (3) using the NN with spatial graph Laplacian in the hidden layer, (4) using the NN with spectral graph Laplacian in the hidden layer, and finally, (5) using the NN with both spectral graph Laplacian in the hidden layer followed by spatial regularization using RW algorithm, which is the complete solution proposed in this paper. These tables also show execution times in seconds, which include both training and testing times.

The results show clearly that semisupervised learning has improved the accuracy significantly for all types of CNN. However, in agreement with an earlier experiment, the best results achieved are with the *vgg-very-deep-16* pretrained CNN, with an OA of 91.23% and an AA of 83.68%. Also, the semisupervised learning has significantly improved the accuracy of the NN as the Z score reached -4.67. Similarly for the Riyadh image, the results shown in Table 3 indicate an even better improvement induced by semisupervised learning with a Z score below -5 for all types of CNN. Again, the best results are achieved using the *vgg-very-deep-16* pretrained CNN, with an OA of 95.26%, an AA of 92.91%, and a Z score of -5.40.

Finally, qualitative results are also shown in Figures 3 and 4 for the Jeddah and Riyadh datasets, respectively. Here, we show the classification maps that have achieved the highest OA after semisupervised learning among the 10 successive runs of the algorithm. In Figures 3(e) and 4(e), we show the tiles whose labels are corrected by semisupervised learning. Comparing Figure 3(d) to Figure 3(c), one can see the type of improvement produced by semisupervised learning, namely, the reduced noise particularly in the water and in residential classes. As for the Riyadh classification maps shown in Figures 4(c) and 4(d), the improvements are clearly visible in the freeway class, where it is more recognizable as roads.

In this final experiment, we compare the proposed method to a state-of-the-art method in the literature called the Laplacian extreme learning machine (LapELM) method [37]. To make the comparison fairer, we have changed the features used in the original method [37] to the same CNN features used here. Thus, the two methods

Table 2: Classification results for the Jeddah dataset using one hidden layer of size 256.

	OA	AA	Time (s)	Z score	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
VGG-m													
NN	86.05	72.84	0.23	0.00	60.32	82.35	54.40	82.38	56.58	73.77	77.03	71.72	96.97
NN + RW	86.42	73.24	0.04	**−1.23**	60.32	81.76	53.60	84.45	53.68	76.56	77.57	73.91	97.28
Spatial graph in the hidden layer	58.24	36.42	19.86	**20.96**	17.30	61.57	18.40	8.36	21.86	52.29	36.49	34.22	77.33
Spectral graph in the hidden layer	87.75	77.41	25.29	**−2.71**	80.00	84.90	57.40	81.77	80.57	59.34	86.22	69.69	96.81
NN + spectral graph + RW	88.33	78.08	0.04	**−3.59**	80.64	84.71	56.80	83.05	81.25	61.97	85.95	71.41	97.01
GoogleNet													
NN	84.48	73.23	0.23	0.00	71.75	87.45	55.20	71.89	75.86	44.81	83.78	72.35	96.00
NN + RW	85.21	74.39	0.04	**−2.44**	72.86	87.45	55.20	75.80	76.94	46.12	84.59	74.38	96.14
Spatial graph in the hidden layer	77.37	58.45	20.05	**6.59**	47.30	87.65	11.60	78.72	70.21	35.57	58.92	45.31	90.81
Spectral graph in the hidden layer	85.97	76.82	26.38	**−2.57**	73.65	91.57	60.60	81.10	87.82	39.07	91.08	70.94	95.55
NN + spectral graph + RW	86.46	77.76	0.04	**−3.32**	74.92	91.77	60.80	84.27	89.33	37.60	91.89	73.59	95.67
ResNet													
NN	87.68	79.33	0.37	**0.00**	81.59	83.14	63.80	91.04	75.13	70.00	81.35	73.59	94.36
NN + RW	88.33	79.69	0.05	**−2.59**	81.27	84.51	62.60	91.28	75.49	70.22	82.70	73.91	95.26
Spatial graph in the hidden layer	78.82	62.59	33.94	**10.27**	46.03	93.34	33.80	82.38	61.40	64.86	39.73	52.97	88.78
Spectral graph in the hidden layer	88.65	82.96	38.38	**−2.00**	84.76	91.96	78.00	91.95	79.22	69.73	86.76	70.16	94.12
NN + spectral graph + RW	89.01	82.92	0.06	**−2.63**	84.29	92.16	76.60	92.44	79.64	69.67	87.30	69.53	94.70
Very-deep-16													
NN	88.41	78.52	0.62	0.00	74.76	88.63	67.60	96.34	57.67	76.45	82.16	65.63	97.45
NN + RW	89.37	78.98	0.09	**−3.77**	80.95	88.24	84.00	93.90	50.26	86.89	81.08	51.56	96.85
Spatial graph in the hidden layer	68.79	56.74	60.98	**16.15**	44.60	67.06	29.60	54.15	67.41	48.64	72.16	49.06	77.97
Spectral graph in the hidden layer	90.28	83.45	96.01	**−3.00**	82.86	90.59	74.80	93.96	74.77	73.28	95.67	68.28	96.86
NN + spectral graph + RW	91.23	83.68	0.09	**−4.67**	84.13	90.59	72.40	94.39	74.51	74.48	95.13	69.22	98.25

now use the same CNN features extracted with the help of the vgg-very-deep-16 pretrained CNN. As for the other parameters of the LapELM method, we have set the ranges for the regularization parameter C to [0.001, 1000] and the RBF kernel parameter γ to [0.01, 5]. To estimate the best parameter values in these ranges, we use the deferential evolution technique, which outperforms the grid search technique as shown in [37]. As for the proposed semisupervised method, we note that the batch size parameter is usually set to 100 in the literature. However, here, we should take care not to set the batch size to be larger than the total number of training samples (number of samples per class × the number of classes). For this experiment, we set it equal to the number of samples per class times the number of classes.

We also study the sensitivity of the method to reduced training samples (less than 10). Figure 8 shows the OA for sample sizes from 2 to 10 for both methods being compared. Figure 8(a) shows the results for the Jeddah dataset, while Figure 8(b) shows the result for the Riyadh dataset. The first observation is that the classification accuracy for both methods degrades gracefully when the number of training samples per class decreases. However, the results confirm again the superiority of the proposed method compared to Laplacian ELM.

4. Conclusions and Future Work

In this letter, we proposed a practical and efficient solution to the semisupervised classification of large-scale VHR RS

Table 3: Classification results for the Riyadh dataset using one hidden layer of size 256.

	OA	AA	Time (s)	Z score	Class 1	Class 2	Class 3	Class 4	Class 5
VGG-m									
NN	87.96	81.98	0.22	—	99.09	71.49	86.26	96.53	56.55
NN + RW	88.07	82.21	0.05	**−0.49**	99.15	71.37	87.27	96.56	56.67
Spatial graph in the hidden layer	81.52	71.48	24.00	**6.25**	81.90	74.57	78.79	94.33	27.82
Spectral graph in the hidden layer	93.12	88.98	30.36	**−5.30**	99.22	85.03	91.01	98.61	71.04
NN + spectral graph + RW	93.18	89.07	0.05	**−5.38**	99.22	85.09	91.52	98.64	70.92
GoogleNet									
NN	89.03	88.63	0.37	—	94.51	85.89	82.22	89.48	91.03
NN + RW	89.43	89.05	0.08	**−1.06**	94.51	85.43	83.64	90.09	91.61
Spatial graph in the hidden layer	80.97	75.54	23.97	**5.76**	96.21	77.77	70.61	85.50	47.59
Spectral graph in the hidden layer	93.45	91.03	55.17	**−4.80**	97.91	90.34	87.98	96.26	82.64
NN + spectral graph + RW	93.84	91.40	0.09	**−5.20**	97.91	90.40	88.18	96.83	83.68
ResNet									
NN	89.58	86.33	0.35	—	88.30	81.09	87.78	95.40	79.08
NN + RW	89.72	86.44	0.07	**−0.50**	89.22	81.26	86.97	95.46	79.31
Spatial graph in the hidden layer	73.64	65.76	39.26	**8.61**	99.68	30.91	90.20	87.19	20.81
Spectral graph in the hidden layer	95.13	93.56	45.94	**−5.86**	98.76	89.54	96.36	97.50	85.63
NN + spectral graph + RW	95.29	93.80	0.07	**−6.00**	99.09	89.43	96.47	97.61	86.44
Very-deep-16									
NN	90.97	88.09	0.63	—	95.36	75.60	94.85	96.81	77.82
NN + RW	93.13	91.84	0.11	**−1.71**	97.39	82.29	95.96	96.22	87.36
Spatial graph in the hidden layer	79.73	67.40	69.42	**6.74**	80.65	62.29	83.64	97.19	13.22
Spectral graph in the hidden layer	94.72	92.21	91.27	**−4.72**	98.00	88.85	95.12	98.29	80.79
NN + spectral graph + RW	95.26	92.91	0.11	**−5.40**	98.50	90.06	94.45	98.55	82.99

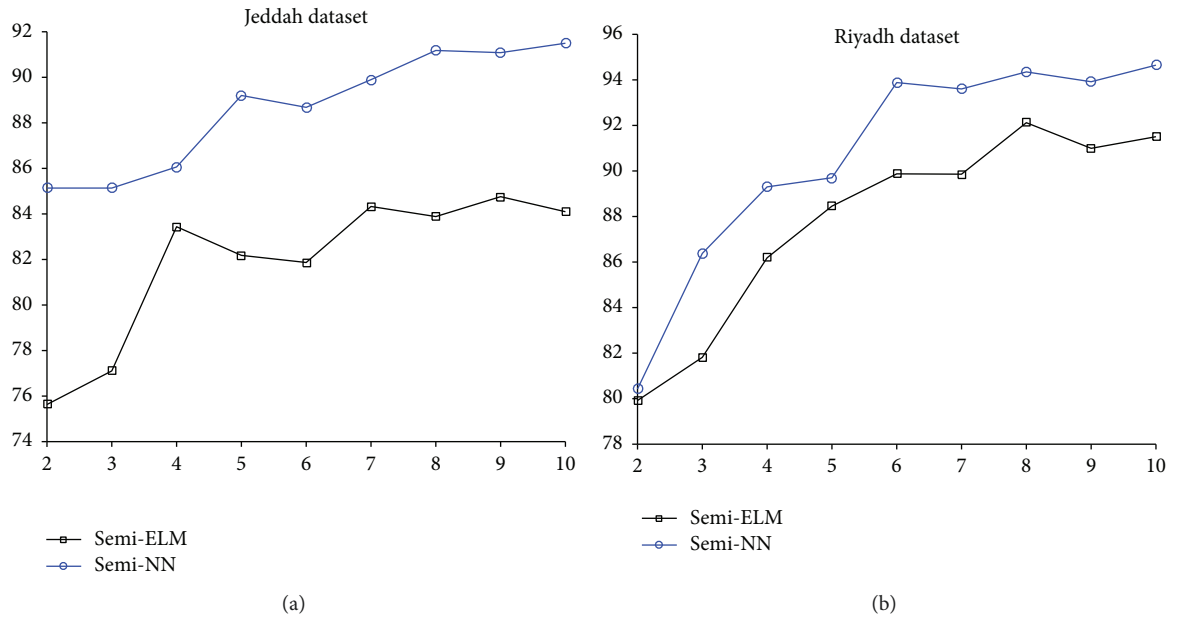

Figure 8: Comparison between the proposed semisupervised NN and the Laplacian ELM method. (a) Results for Jeddah dataset. (b) Results for Riyadh dataset.

images. The proposed method is based on subdividing the image into square tiles and then extracting feature descriptors on these tiles. These in turn are fed into another fully connected neural network with semisupervised learning, which exploits the structural information contained in the unlabelled tiles. Semisupervised learning is implemented in our proposed method in two ways, first by embedding a graph Laplacian in the hidden layer and second by spatial regularization of the output layer using a random walker algorithm.

Experimental results have shown that our proposed method can provide coarse classification maps for very large RS images with high accuracy. Furthermore, we have shown that our proposed semisupervised learning that combines a graph Laplacian in the hidden layer and spatial regularization of the output layer always provides significant improvement in terms of the overall accuracy, while having reasonable computational times for such large VHR images.

As a future research direction, we can employ a self-paced selection strategy of unlabelled titles similar to the techniques used in [38, 39] to further improve the performance. One can also study different fusion techniques of the feature descriptors of many pretrained CNNs. Another direction is to research ways to combine data from different datasets in order to improve the accuracy of the learning algorithm.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

The authors would like to extend their sincere appreciation to Deanship of Scientific Research at King Saud University for funding this research group (no. RG-1435-050).

References

[1] P. Lassalle, J. Inglada, J. Michel, M. Grizonnet, and J. Malik, "A scalable tile-based framework for region-merging segmentation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 10, pp. 5473–5485, 2015.

[2] T. Moranduzzo, F. Melgani, M. L. Mekhalfi, Y. Bazi, and N. Alajlan, "Multiclass coarse analysis for UAV imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 12, pp. 6394–6406, 2015.

[3] T. Moranduzzo, M. L. Mekhalfi, and F. Melgani, "LBP-based multiclass classification method for UAV imagery," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pp. 2362–2365, Milan, Italy, 2015.

[4] X. Yao, J. Han, G. Cheng, X. Qian, and L. Guo, "Semantic annotation of high-resolution satellite images via weakly supervised learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 6, pp. 3660–3671, 2016.

[5] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10*, pp. 270–279, New York, NY, USA, 2010, ACM Press.

[6] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pp. 886–893, San Diego, CA, USA, 2005.

[8] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[9] K. Fukushima, "Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[10] Y. LeCun, B. Boser, J. S. Denker et al., "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[11] M. Hayat, M. Bennamoun, and S. An, "Learning non-linear reconstruction models for image set classification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1915–1922, Columbus, OH, USA, 2014.

[12] J. Bai, Y. Wu, J. Zhang, and F. Chen, "Subset based deep learning for RGB-D object recognition," *Neurocomputing*, vol. 165, pp. 280–292, 2015.

[13] S. Gao, Y. Zhang, K. Jia, J. Lu, and Y. Zhang, "Single sample face recognition via learning deep supervised autoencoders," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 10, pp. 2108–2118, 2015.

[14] T. Brosch and R. Tam, "Efficient training of convolutional deep belief networks in the frequency domain for application to high-resolution 2D and 3D images," *Neural Computation*, vol. 27, no. 1, pp. 211–227, 2015.

[15] G. Hinton, L. Deng, D. Yu et al., "Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[16] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 865–873, 2015.

[17] "Visual geometry group home page," May 2017, http://www.robots.ox.ac.uk/~vgg/research/very_deep/.

[18] A. Anuse and V. Vyas, "A novel training algorithm for convolutional neural network," *Complex & Intelligent Systems*, vol. 2, no. 3, pp. 221–234, 2016.

[19] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," Computing Research Repository CoRR 2015, http://arxiv.org/abs/1508.00092.

[20] D. Marmanis, M. Datcu, T. Esch, and U. Stilla, "Deep learning earth observation classification using ImageNet pretrained networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 1, pp. 105–109, 2016.

[21] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 10, pp. 1797–1801, 2014.

[22] F. Zhang, B. Du, and L. Zhang, "Scene classification via a gradient boosting random convolutional network framework,"

IEEE Transactions on Geoscience and Remote Sensing, vol. 54, no. 3, pp. 1793–1802, 2016.

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

[24] G. Cheng, J. Han, and X. Lu, "Remote sensing image scene classification: benchmark and state of the art," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1865–1883, 2017.

[25] Y. Bazi, N. Alajlan, and F. Melgani, "Improved estimation of water chlorophyll concentration with semisupervised Gaussian process regression," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 7, pp. 2733–2743, 2012.

[26] X. Yin, W. Yang, G. S. Xia, and L. Dong, "Semi-supervised feature learning for remote sensing image classification," in *2014 IEEE Geoscience and Remote Sensing Symposium*, pp. 1261–1264, Quebec City, QC, Canada, 2014.

[27] Y. Bazi and F. Melgani, "Semisupervised PSO-SVM regression for biophysical parameter estimation," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 6, pp. 1887–1895, 2007.

[28] L. Bruzzone and C. Persello, "Recent trends in classification of remote sensing data: active and semisupervised machine learning paradigms," in *2010 IEEE International Geoscience and Remote Sensing Symposium*, pp. 3720–3723, Honolulu, HI, USA, 2010.

[29] Z. Wang, N. M. Nasrabadi, and T. S. Huang, "Semisupervised hyperspectral classification using task-driven dictionary learning with Laplacian regularization," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 3, pp. 1161–1173, 2015.

[30] O. Chapelle, B. Schlkopf, and A. Zien, *Semi-Supervised Learning*, The MIT Press, 1st edition, 2010.

[31] J. Weston, F. Ratle, and R. Collobert, "Deep learning via semi-supervised embedding," in *Proceedings of the 25th International Conference on Machine Learning - ICML '08*, pp. 1168–1175, New York, NY, USA, 2008, ACM.

[32] L. Grady, "Random walks for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1768–1783, 2006.

[33] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: delving deep into convolutional nets," Computing Research Repository CoRR 2014, http://arxiv.org/abs/1405.3531.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Computing Research Repository CoRR 2014, http://arxiv.org/abs/1409.1556.

[35] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," Computing Research Repository CoRR 2014, http://arxiv.org/abs/1409.4842.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Computing Research Repository CoRR 2015, http://arxiv.org/abs/1512.03385.

[37] Y. Bazi, N. Alajlan, F. Melgani, H. AlHichri, S. Malek, and R. R. Yager, "Differential evolution extreme learning machine for the classification of hyperspectral images," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 6, pp. 1066–1070, 2014.

[38] X. Yao, J. Han, D. Zhang, and F. Nie, "Revisiting co-saliency detection: a novel approach based on two-stage multi-view spectral rotation co-clustering," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3196–3209, 2017.

[39] D. Zhang, D. Meng, and J. Han, "Co-saliency detection via a self-paced multiple-instance learning framework," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 5, pp. 865–878, 2017.