remote sensing

MDPI

Article

Detection of Informal Settlements from VHR Images Using Convolutional Neural Networks

Nicholus Mboga, Claudio Persello *, John Ray Bergado and Alfred Stein

Faculty of Geo-Information Science and Earth Observation (ITC), University of Twente, 7500 AE Enschede, The Netherlands; n.o.mboga@student.utwente.nl (N.M.); j.r.bergado@utwente.nl (J.R.B.); a.stein@utwente.nl (A.S.)

* Correspondence: c.persello@utwente.nl; Tel.: +31-534874343

Received: 30 September 2017; Accepted: 25 October 2017; Published: 30 October 2017

Abstract: Information about the location and extent of informal settlements is necessary to guide decision making and resource allocation for their upgrading. Very high resolution (VHR) satellite images can provide this useful information, however, different urban settlement types are hard to be automatically discriminated and extracted from VHR imagery, because of their abstract semantic class definition. State-of-the-art classification techniques rely on hand-engineering spatial-contextual features to improve the classification results of pixel-based methods. In this paper, we propose to use convolutional neural networks (CNNs) for learning discriminative spatial features, and perform automatic detection of informal settlements. The experimental analysis is carried out on a QuickBird image acquired over Dar es Salaam, Tanzania. The proposed technique is compared against support vector machines (SVMs) using texture features extracted from grey level co-occurrence matrix (GLCM) and local binary patterns (LBP), which result in accuracies of 86.65% and 90.48%, respectively. CNN leads to better classification, resulting in an overall accuracy of 91.71%. A sensitivity analysis shows that deeper networks result in higher accuracies when large training sets are used. The study concludes that training CNN in an end-to-end fashion can automatically learn spatial features from the data that are capable of discriminating complex urban land use classes.

Keywords: image classification; informal settlements; convolutional neural networks; deep learning; very high resolution satellite imagery

1. Introduction

Rapid urbanization has led to the proliferation of informal settlements in developing countries. While a clear definition of what constitutes an informal settlement is missing, it usually refers to an area where the land tenure is not recognized by the public authorities [1], and a neighborhood where the residents have sub-standard housing and insufficient basic services [2]. Several terms have been used to allude to informal settlements in urban contexts around the world, for example "squatter settlements", "favelas", "shacks", "bajos", "bidonvilles" and "slums" [3]. Informal settlements usually contain unplanned settlements that are developed disregarding zoning, land use plans and service allocation [4,5]. There is a shortage of spatial information regarding such informal settlements, which is necessary for decision making and planning the activities for their improvement. Informal settlements grow fast, and mostly cover large extents in an urban area, or in some instances form scattered pockets within formal settlement areas. There is a need for classification methods that are able to provide spatial information in a timely and accurate way [6].

The availability of very high resolution (VHR) satellite images has provided the capability to acquire spatial information about informal settlements. The advantage of these images is that they cover a wide geographical area, and enable characterization of urban settlement types based on

their morphological characteristics. Informal settlements mostly comprise small, clustered buildings and little vegetation, whereas formal areas consist of larger buildings arranged in a regular spatial pattern and vegetation [7]. Two issues are important: (i) a high spatial resolution implies high within-class variance and low between-class variance and (ii) urban settlement types, such as informal settlements, are associated with a higher level of semantic abstraction as opposed to land cover classes. The extraction of spatial-contextual features is thus needed to improve the classification of VHR images to obtain the desired land use classes [8,9]. While spatial information refers to the spatial arrangement of spectral information in a scene, contextual information describes the information extracted from a neighborhood of a particular pixel in the image [10].

Standard methods for extracting spatial-contextual features based on hand-engineering have been used for the detection of informal settlements from VHR images. These include morphological profiles [11], GLCM features [12] and lacunarity [13]. Moreover, a comparison of various texture features such as local binary patterns (LBPs), histogram of oriented gradients (HOGs) and line support regions (LSRs), among others, is done in [14,15]. A limitation is that extraction of a specific feature depends on the particular technique used and that their performance on the data is not known beforehand. In addition, they have free parameters that need to be optimized by a user through trial and error [16]. In this work, we propose to use CNNs to automate the extraction of spatial-contextual features by learning them directly from the data [17].

CNNs are artificial neural networks that learn spatial-contextual features in several hierarchical nonlinear layers [17]. They have been studied widely for computer vision and speech recognition tasks [18]. CNNs have performed well in multimedia image classification, such as in the ImageNet LSVRC-2010 contest [19] and in computer vision benchmarks [20]. In recent times, there have been some applications in land cover classification from satellite imagery [9,21–25], as well as in land use classification [26–29]. Land use classes, such as informal settlements, are more complex because they can contain more than one type of land cover, at different scales and orientations.

The aim of this work is to investigate deep CNNs to detect informal settlements from VHR satellite imagery. Using a QuickBird image acquired over Dar es Salaam, Tanzania, the CNN was trained in an end-to-end way and its performance of using learned spatial-contextual features was analyzed. The results were then compared with the performance of state-of-the-art classifier relying on hand-engineered features. The remainder of the paper is organized as follows: Section 2 describes the adopted methods, Section 3 describes the experimental set up, Section 4 the results and analysis, Section 5 the discussion and Section 6, the conclusion.

2. Method

2.1. CNN Building Blocks

We developed and optimized a dedicated CNN based on the general architecture shown in Figure 1. The architecture consists of g convolutional layers responsible for learning the spatial-contextual features from the image, and t fully connected layers that derive the discriminative function using softmax activations. During the training phase, patches with a fixed size are used as input to the CNN to learn the information class of the central pixel of the patch [19]. At inference time, a sliding window produces a pixel-wise classification map [9]. The size of the sliding window is determined by the size of the input patches used during the training.

CNNs make use of convolutions in at least one of their layers. A convolutional layer contains k filters of size $h \times h \times b$, where h is the height and width of the filter and b is the number of input bands. The first convolutional layer performs a convolution over the 3-D input volume of dimension $m \times m \times b$, where m defines the spatial dimension of the patch. The output of the convolutional layer has a dimension of $\frac{m-k+2z}{s} \times \frac{m-k+2z}{s} \times k$, where z is the number of zeros used to pad the input, k is the dimension of the response activations, and s is the stride of the convolution, i.e., the number of steps by which the filter is shifted during the convolution.

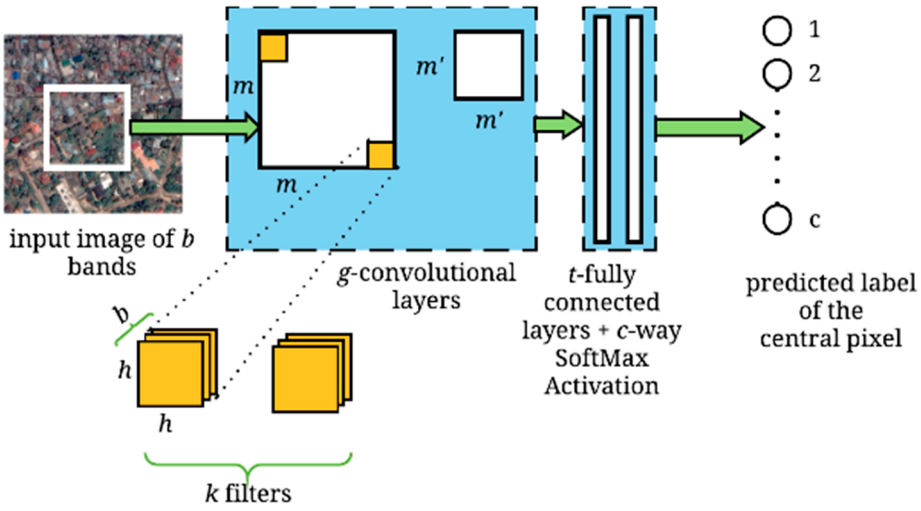

Figure 1. Overview of a patch-based convolutional neural networks (CNN) architecture.

Zero padding consists adding a number of rows and columns of zeros to the border of the input image patch to control the size of the obtained feature map. In the convolutional layers, using a filter with a size less than the input reduces the number of connections, hence the number of parameters when determining the output. Moreover, the same set of weights is learned for each location in the input image for a particular convolutional layer, resulting in parameter sharing [30].

The response activations from a convolutional layer are linear in nature. They are passed through a nonlinear activation function, resulting in a nonlinear transformation. Saturating nonlinearities such as the hyperbolic tangent, $f(x) = \tan \mathrm{h}(x)$ and the sigmoidal function, $f(x) = (1 + e^{-x})$ suffer from the vanishing gradient problem [31,32]. This means that during training, the gradient of the output of the network with respect to its parameters tend to remain extremely small even if inputs with a large magnitude are used, thereby the learning algorithm lacks the guide to update the parameters. Un-saturating nonlinear activations such as the rectified linear unit (RELU) are robust to this problem [32]. RELU is given as $f(x) = \max(0, x)$ and is useful in optimizing models that are gradient based because they remain mostly linear. Faster training of networks is observed if RELU nonlinearity is used as compared to saturating units such as the hyperbolic tangent and the sigmoidal function [33].

Pooling is the use of summary statistics of adjacent outputs in a feature map to determine the activations to be propagated to the subsequent layer. It results in rotationally and translationally invariant features [34]. For a window of size $p \times p$, average pooling returns the arithmetic mean of signals in the window whereas maxpooling returns the dominant signal in that window. The stride of a pooling layer refers to the number of steps between two subsequent pooling windows. Pooling with a stride $s > 1$ results into downsampling with a factor s. Pooling with downsampling reduces the spatial dimension of the output [35]. Effectively, the computation cost decreases. Also, the receptive field of the subsequent convolutional layers increases as more convolutional layers with down-sampling are used.

The output of the final convolutional layer is flattened to a vector containing the extracted features and fed into the fully connected layer with l neurons. The output of the last fully connected layer is normalized using a softmax activation function. It has c units, representing the number of classes and gives the class distribution scores for each label y_i expressed as:

$$p(y_i|x_i) = \frac{\exp(x_i)}{\sum_{j=1}^{c} \exp(x_j)} \quad (1)$$

where x_i is an input vector representing the un-normalized scores for the sample i for $i = 1 \ldots c$. The parameters of the network are determined by means of supervised training by minimizing the negative log likelihood over the training data. The loss function is equal to:

$$L(w) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C} y_{nc}\log(\hat{y}_{nc}) \tag{2}$$

where w represents the weights (i.e., the parameters of the network), y_{nc} is the vector of the true labels and $\hat{y}_{nc}$ is the vector of the predicted labels for a training set comprising of N samples. The loss function quantifies the misclassification by comparing y_{nc} and $\hat{y}_{nc}$ for N training samples. The CNN is trained using stochastic gradient descent (SGD) with momentum α and learning rate ϵ from a small subset of training data called a mini-batch [36]. The decay learning rate, ϵ_d is used to model the random noise introduced by the SGD that is present even after the loss function is minimized. These parameters influence the ability of the SGD to minimize the loss function [31]. The parameters of the network, w are learned using backpropagation with SGD. The weights are updated in Equation (3) while the learning rate is updated using Equation (4):

$$\Delta w(\tau) = -\epsilon(\tau)\frac{\partial L_\tau}{\partial w} + \alpha\Delta w(\tau - 1) \tag{3}$$

$$\epsilon(\tau) = \frac{\epsilon_0}{1 + \epsilon_d \tau} \tag{4}$$

where Δw are the weight updates, $\frac{\partial L_\tau}{\partial w}$ are the partial derivatives, and ϵ_0 is the initial learning rate. τ and $\tau - 1$ represent the current and previous epochs of the training phase [9,37].

During training, the CNN is likely to overfit because of limited training data. One way to mitigate this is by means of dropout, where a percentage d_r of the neurons and their connections is turned off [38]. A second way is to use early stopping: the algorithm is run until the error on the validation set does not improve for a given number of epochs, e_n. A third way is to penalize the parameters deviating from zero. The resulting cost function after adding ℓ_2-norm regularization is given as:

$$L_1(w) = L(w) + \lambda \parallel w \parallel_2^2 \tag{5}$$

where λ is the regularization parameter, and $\parallel w \parallel_2$ is the ℓ_2-norm of the weight vector.

2.2. Adopted Architecture

Our proposed network comprises convolutional layers, fully connected layers and a softmax classification layer. In each convolutional layer, a series of operations is performed on the input: (i) a 2-D convolution, (ii) point-wise nonlinear transformation using RELU and (iii) maxpooling with subsampling. A schematic overview of a convolutional layer of the proposed CNN architecture is shown in Figure 2.

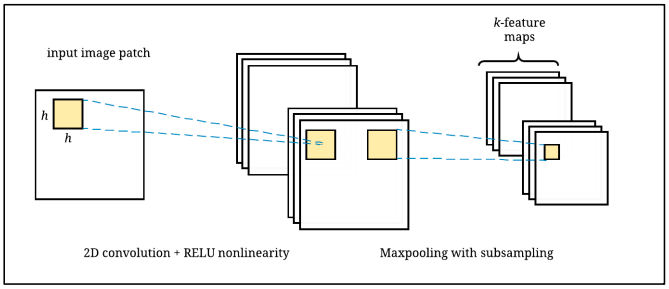

Figure 2. An illustration of the convolutional layer of the proposed architecture. The receptive fields have a dimension of $h \times h$.

The fully connected layers summarize the learned features and the softmax activation gives the distribution of scores for each class. We carried out experiments varying the structure of the network. An overview of the hyperparameters is presented in Table 1. The training hyperparameters are kept constant while different values of the spatial feature learning hyperparameters are tried.

Table 1. Overview of CNN hyperparameters.

Training Hyperparameters	Spatial Feature Learning Hyperparameters
Learning rate, ϵ	Patch size, m
Momentum, α	Convolutional layers, g
Learning rate decay, ϵ_d	Fully connected layers, t
Early stopping patience,	Number of filters, k
Maximum number of epochs, e_n	Filter size, h
Weight decay, λ	
Dropout rate, d_r	

3. Experimental Set-Up

3.1. Data

We use a QuickBird image acquired over Dar es Salaam, Tanzania, acquired in 2007 (Figure 3) for the experiments. The city of Dar es Salaam is considered because about 80% of its buildings are located in informal areas, and 80% of the residents of Dar es Salaam live in informal areas [1].

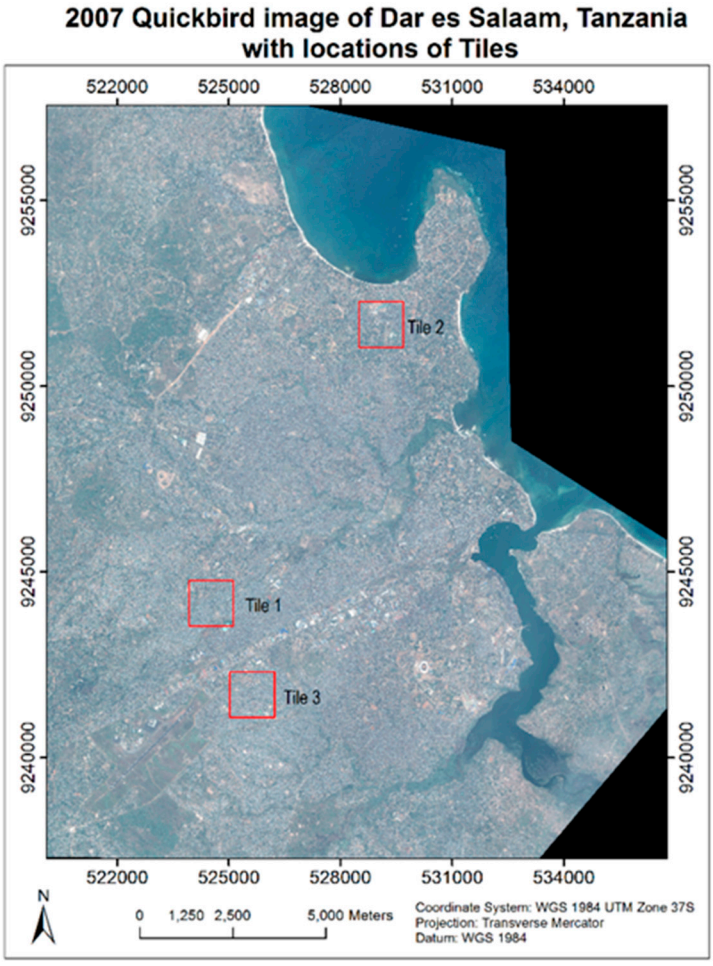

Figure 3. A map showing location of the tiles used for the experiments i.e., Tile 1, Tile 2 and Tile 3.

The multispectral image is pan-sharpened with a spatial resolution of 0.60 m. It has four bands, namely, blue, green, red and near infrared. A reference land use vector map prepared in 2004 is available [39,40], and has four classes: "informal", "formal", "other urban" and "agricultural". It was used, together with visual image interpretation, to prepare the reference data to be used in the experiments. To carry out a binary classification, a class "other" was created by combining the last three classes, while maintaining the "informal" class. This was done to evaluate the ability of the classifier to distinguish the "informal" settlement class from all other urban classes. Three tiles of 2000 × 2000 pixels, covering an area of 1.2 × 1.2 km on the ground were used. The raw image tiles and the corresponding reference data are presented in Figure 4.

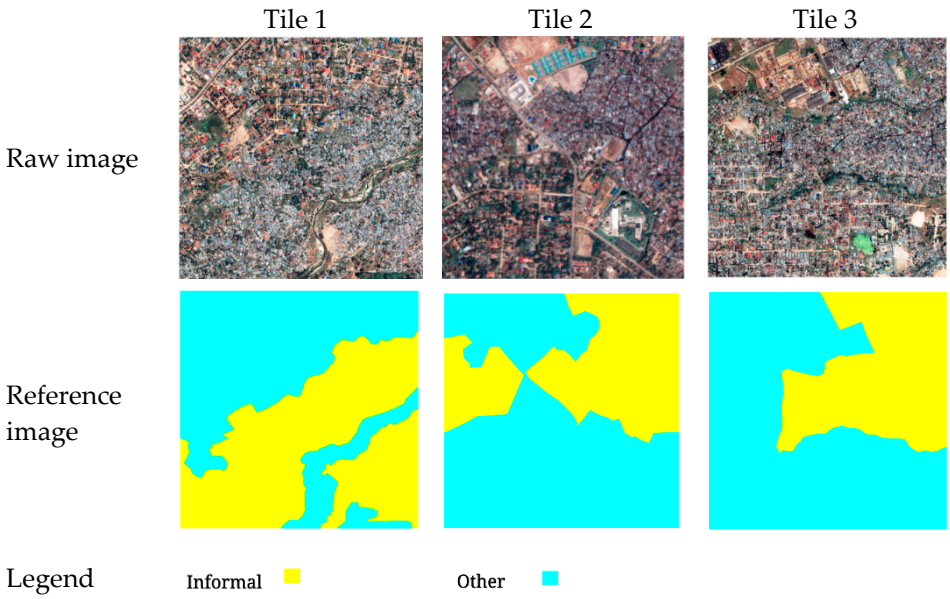

Figure 4. Raw QuickBird images of Dar es Salaam, Tanzania, and the corresponding reference data for Tile 1, Tile 2 and Tile 3 prepared by means of visual image interpretation. Each tile covers an area of 1.2 × 1.2 km on the ground.

The Theano deep learning framework and the Keras library were used for designing the CNN, Python language for programming and R for extracting GLCM features [41–43].

3.2. Training Hyperparameters

The rate at which the SGD converges, and whether it is at an optimal solution is influenced by the training hyperparameters. Preliminary experiments using a subset of 500 × 500 pixels showed a stable classification accuracy upon varying these hyperparameters. The hyperparameter values kept constant during the CNN experiments are: $\epsilon = 0.001$, $\alpha = 0.9$, $\epsilon_d = 0.001$, $e_n = 50$, $\lambda = 0.0001$. A dropout rate, $d_r = 0.5$ is used in the fully connected layers. The CNN configuration is shown in Table 2.

Table 2. CNN configuration.

Parameters	Values
Layers [1]	I − (C – A – P − D1) × g − (F−D2) × t − O
Nonlinearity used in A and F	RELU
Nonlinearity used in O	softmax
Pooling size	2
Width of F	128

[1] Layer Notation: I = input, C = convolutional layer, A = Activation function, P = pooling, F = fully connected layer and O = output. The convolutional stride is 1.

A mini-batch size of 128 is used. Weights are initialized using normalized initialization [32]. We exclude $\frac{m}{2} - 1$ border pixels from all sides of the tile when selecting samples during training and testing, where m is the patch size in pixels. This is to avoid likely misclassification resulting from padding the border pixels with zeros when selecting the samples.

3.3. Spatial Feature Learning Hyperparameters

We investigate spatial feature learning hyperparameter values for an optimally designed CNN. They include the patch size, m = [65, 99, 129, 165], the number of convolutional layers, g = [2, 3, 4] the number of the fully connected layers, t = [1, 2, 3], the number of the filters, k = [8, 16, 32, 64] and the kernel dimension, h = [7, 17, 25]. We keep the patch size constant at m = 99 while optimizing the hyperparameters in order to keep a low computational cost. The network is trained using SGD over a sample set of size 2160. The accuracy is calculated using an independent test set drawn from the three tiles. A summary of the experimental values is presented in Table 3.

Table 3. A summary of the values used in CNN hyperparameter optimization. The columns represent the experiment carried out. The main diagonal shows the values tried.

Parameters	Patch Size m	Convolutional Layers g	Fully Connected t	Number of Filters k	Filter Size h
Patch size m	(65,99,129,165)	99	99	99	99
Convolutional layers g	2	(2,3,4)	2	2	2
Fully connected	1	1	(1,2,3)	1	1
Number of filters k	8	8	8	(8,16,32,64)	8
Filter size h	7	7	7	7	(7,17,25)

Spatial feature learning hyperparameters affect the quality of the spatial-contextual features that are automatically extracted during training. The patch size defines the size of area from which contextual information is derived and influences the assigned label [44]. The number of kernels (filters) denotes the variety of spatial patterns that can be learned to distinguish the land use classes. The dimension of the kernel defines the size of the patterns considered when distinguishing between classes of interest. Fully connected layers make use of all the features in the previous convolutional layers to derive the final classification rule [9].

3.4. Baseline Method: SVM with GLCM Features and LBP Features

As a comparison, we use SVM with RBF kernel making use of GLCM features. Spatial contextual features are extracted from the GLCM and used in the classification [10]. GLCM variance was shown to be useful in the detection of informal settlements in [12] and is obtained as:

$$f = \sum_i \sum_j (i - \mu)^2 p(i, j) \tag{6}$$

where $p(i, j)$ is the (i, j)th entry in a normalized gray-tone spatial dependence matrix, i and j are gray tones of neighboring pixels. The average of GLCM variance extracted over four directions is used for window size, w_s = [65, 99, 129, 165]. SVM classifiers maximize the margin between the classes while minimizing the training error. It is characterized by high generalization ability (i.e., smaller error on the independent test set) and high classification accuracy [45]. For training the SVM, we use hold out cross validation to determine the regularization parameter C from 1 to 10^3 and the spread of the RBF kernel Y from 10^{-4} to 1 which are logarithmically spaced. The number of border pixels excluded from an image tile is given by $\frac{w_s}{2} - 1$ where w_s is the window size in pixels. During GLCM extraction, the border pixels are padded with zeros, likely resulting to misclassification.

LBP features [46] are extracted and used for comparison, since it was demonstrated in [14] that they could lead to a high accuracy in the classification of urban settlements. Let us define a set of

interpolation points (pixels) P, lying on a circle of radius R. The gray levels of these points is g_p, while that of the center of the circle is g_c. Binary LBP codes are calculated by obtaining the difference between the gray level of the points on the circle and the gray level of the point at its center. A value of 1 is assigned when the difference is positive and 0 when the difference is negative giving a binary sequence illustrated as:

$$T = t(s(g_0 - g_c), s(g_1 - g_c), \ldots, (g_{P-1} - g_c)) \tag{7}$$

where

$$s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{8}$$

By assigning a binomial factor 2^p for each sign $s(g_p - g_c)$, a unique $LBP_{P,R}$ number is obtained that characterizes the local image texture and is defined as:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{9}$$

A histogram of uniform features is computed over a local window of rotation invariant features and is defined in [47] as:

$$LBP_{P,R}^{riu2} = \begin{cases} \sum_{P=0}^{P-1} s(g_p - g_c), & U(LBP_{P,R}) \leq 2 \\ P + 1, & otherwise \end{cases} \tag{10}$$

where

$$U(LBP_{P,R}) = |s(g_{p-1} - g_c) - s(g_0 - g_c)| + \sum_{p=1}^{P-1} \lceil s(g_p - g_c) - s(g_{p-1} - g_c) \rceil \tag{11}$$

The histogram of features is used as the final feature vector. For more information about LBP, we refer the readers to [47]. In this work, LBP features are extracted using R = 3, and P = 8. The histogram of uniform features is computed from the LBP codes using a window of 165 × 165 pixels.

3.5. Size of the Training and Test Set

The size of the training set affects optimal determination of the CNN parameters. Three different training sets of size 1080, 2160 and 3060 were obtained through stratified random sampling based on the frequency of the classes. A patch of size 165 was used because it facilitated the testing of a deep CNN. The convolutional layers were varied between two and six and each had eight kernels with a dimension of 7 × 7. Each training sets is drawn from the three reference maps by stratified random sampling based on the frequency of the two classes. The test set is obtained from the reference maps by excluding training pixels. The resulting number of test pixels used to compute the classification accuracy is equal to 10,116,974. Visual quality assessment of the classified maps is also carried out.

4. Results and Analysis

4.1. Spatial Feature Learning Hyperparameters

In Figure 5a, the analysis of the effect of the patch size shows that high classification accuracy is achieved when a large contextual window is used. Using m = 129 results to an accuracy of 83.29%. This is according to the expectations, because a large contextual neighborhood is needed to discriminate the two abstract land use classes. A decrease in accuracy to 82.87% for m = 165 is because of the limiting factor of using a fixed training sample set of 2160, while the number of parameters grows. It is observed from Figure 5b that using more convolutional layers increases the classification accuracy. Using more layers increases the level of abstraction of the features learned from simple to more complex features.

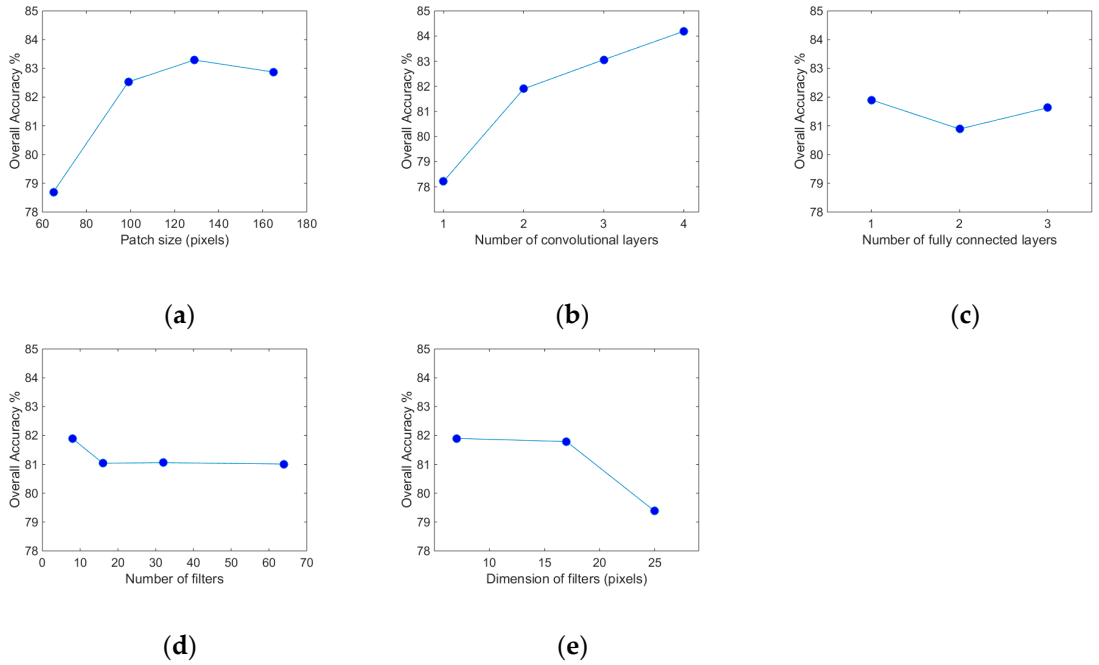

Figure 5. Results of the spatial feature learning hyperparameter optimization for (**a**) patch size, (**b**) convolutional layers, (**c**) fully connected layers, (**d**) number of kernels and (**e**) dimension of kernels.

The designed CNN has a pooling layer with subsampling with a factor of two. The spatial resolution of the feature maps is reduced by two after each convolutional layer. This results in fewer parameters because the number of connections in the fully connected layers is effectively reduced. As seen in Figure 5c, varying the number of fully connected layers has a less significant impact on the classification accuracy.

In Figure 5d, there is a slight decrease in the classification accuracy that is attributed to an increase in the number of parameters as the number of kernels increases. The slight drop is because a larger variety of informative features which contribute to better discrimination of the classes are learnt. A large kernel implies a large receptive field. Consequently, the number of parameters to be optimized by the model during training increases. We observe in Figure 5e that increasing h causes a decrease of 2.5% in accuracy. This is due to an increase in the number of parameters if a large filter dimension is used.

The number of the convolutional layers and the patch size were the most sensitive hyperparameters in our experiments. We also observed that the size of the training set was a limiting factor. Therefore, an experiment to determine the relationship between varying the number of convolutional layers against a training sample was carried out.

4.2. Size of the Training Set

In Figure 6, it is observed that the accuracy increases when more convolutional layers are used. When the number of training samples is small, the classification accuracy saturates at a lower number of convolutional layers. This can be attributed to overfitting of the model as its capacity becomes large. The highest classification accuracy is attained when five convolutional layers are used and with a corresponding training set of 3060. A deeper network can reach higher accuracy by learning discriminative features when trained with large training sets. However, a large training sample is accompanied by more training time [31].

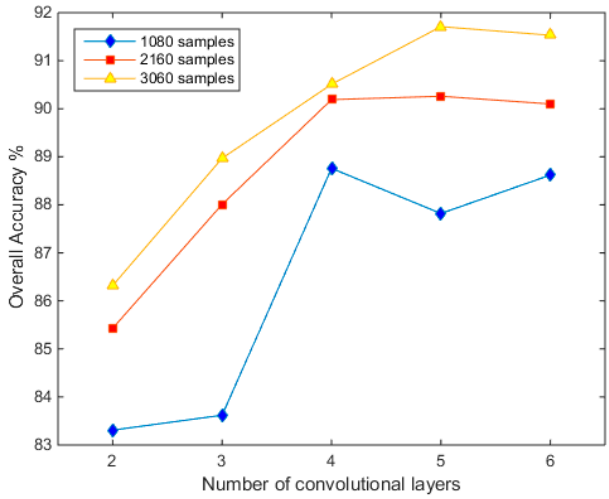

Figure 6. Effect of varying number of convolutional layers while varying the training sample size.

4.3. Classification Results

The classification results from SVM, SVM + GLCM, SVM + LBP and CNN (for g = 2, 3, 4, 5, 6) using a training sample size of 3060 are compared in Table 4. GLCM features extracted using a window size of 165 provided the highest classification accuracy with SVM. Classification relying on spectral features alone results in a low classification accuracy. Addition of spatial-contextual features causes an increase in classification accuracy as seen in SVM + GLCM, SVM + LBP and CNN. Performance of the CNN increases with each addition of a convolutional layer, with the highest classification accuracy being provided by CNN-5. We note that CNN-2 has a lower accuracy than SVM + GLCM. Adding one layer, CNN-3 results in a higher classification accuracy for Tile 2 and Tile 3 only. For g = 4, 5 or 6, CNN outperforms SVM + GLCM. When g = 6, there is a slight drop in the overall accuracy in Tile 1 and Tile 2 of 0.32% and 0.47% respectively, whereas the classification accuracy of Tile 3 increases by 0.26%. CNN-5 results in more than 10% gain over SVM + GLCM in overall accuracy for Tile 3. The classification accuracy of CNN is consistent across the three tiles as opposed to SVM + GLCM. SVM + LBP performs better in Tile 1, Tile 2 and Tile 3 as compared to SVM + GLCM. In addition, it has a high classification accuracy as close to CNN-4, CNN-5 and CNN-6.

Table 4. Comparison of classification accuracies of Support Vector Machines (SVM), SVM + Grey Level Co-Occurrence Matrix (SVM + GLCM), SVM + Local Binary Patterns (SVM + LBP) and Convolutional Neural Networks (CNN). Overall Accuracy (OA) computed across the three tiles is also provided.

Tile ID	SVM	SVM + GLCM	SVM + LBP	CNN-2	CNN-3	CNN-4	CNN-5	CNN-6
Tile 1	59.48	91.99	90.77	88.06	89.60	92.48	93.05	92.73
Tile 2	78.61	88.40	90.49	88.33	90.11	91.28	92.24	91.77
Tile 3	68.45	79.40	90.18	82.57	87.20	87.78	89.85	90.11
OA	68.84	86.60	90.48	86.32	88.97	90.51	91.71	91.53

Figure 7 presents the classified maps for SVM, SVM + GLCM, SVM + LBP and CNN. Results for CNN-2 and CNN-5 are shown. The columns represent Tiles 1, 2 and 3. The first row shows the ground reference maps while the subsequent rows show the results of SVM using spectral features, SVM with GLCM, SVM with LBP, CNN-2 and CNN-5.

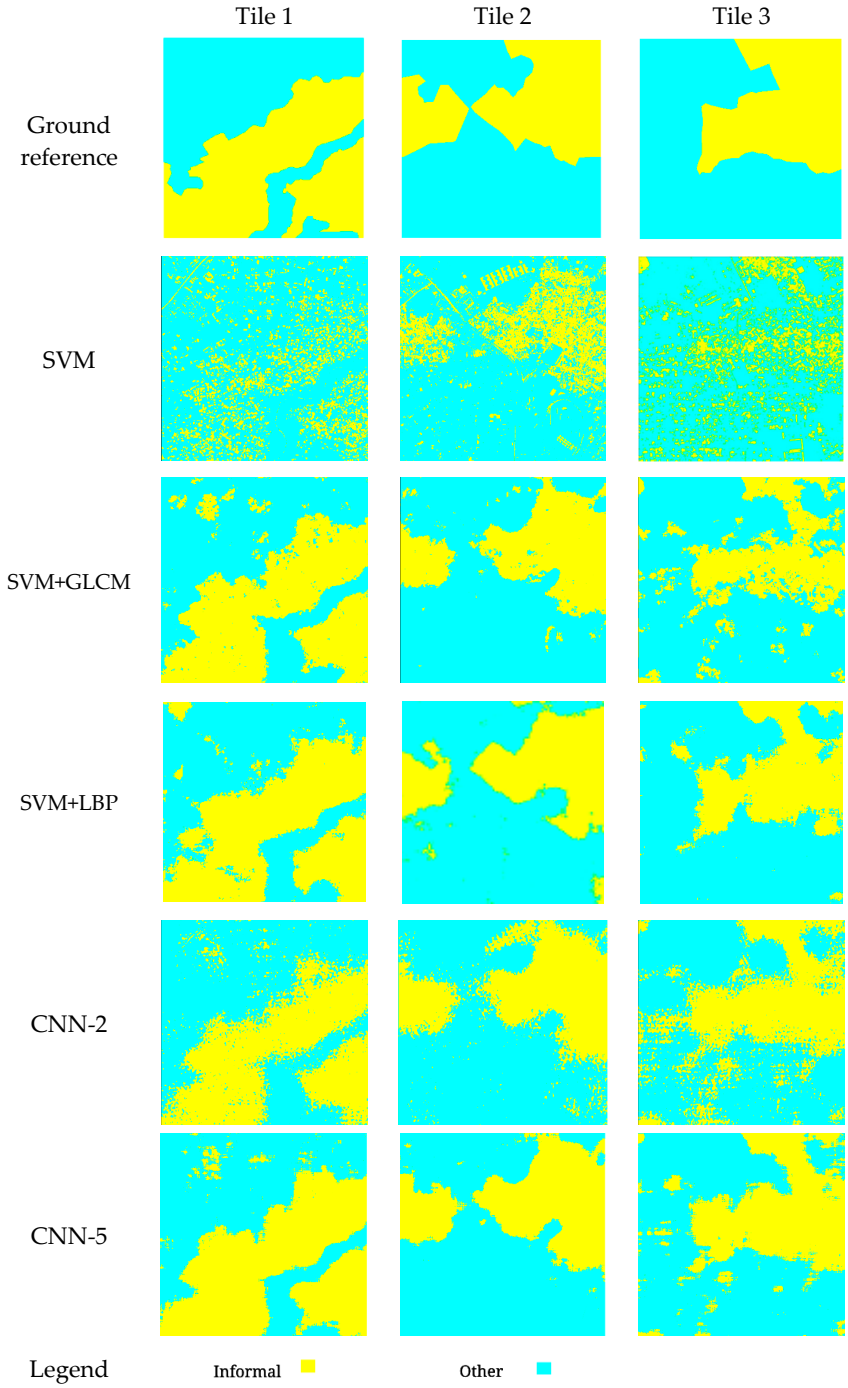

Figure 7. Classification maps from SVM, SVM relying on GLCM features, SVM relying on LBP features and CNN for Tile 1, Tile 2 and Tile 3.

From Figure 7, we observe that maps of SVM using spectral features alone cannot reliably distinguish the two abstract land use classes. The maps where spatial-contextual features have been used have a higher accuracy. The boundary definition is much smoother in CNN-5 as compared to CNN-2. Furthermore, CNN-5 has less noisy classified maps than CNN-2 and some of the misclassification in the northwestern corner of Tile 3 is reduced. From Tile 3, it is evident that the extent of the informal settlement is better captured by CNN approach as opposed to SVM + GLCM approach. We observe that spatial-contextual features that are learned by CNN are capable of discriminating informal settlements from other urban settlement types.

In Figure 8, the northeastern corners of both Tiles 2 and 3 are highlighted. In the input images, these areas are open fields lying within areas of informal settlements. The pixels have been classified as "formal", whereas their corresponding label in the reference image is "informal". This has an effect on the accuracy assessment.

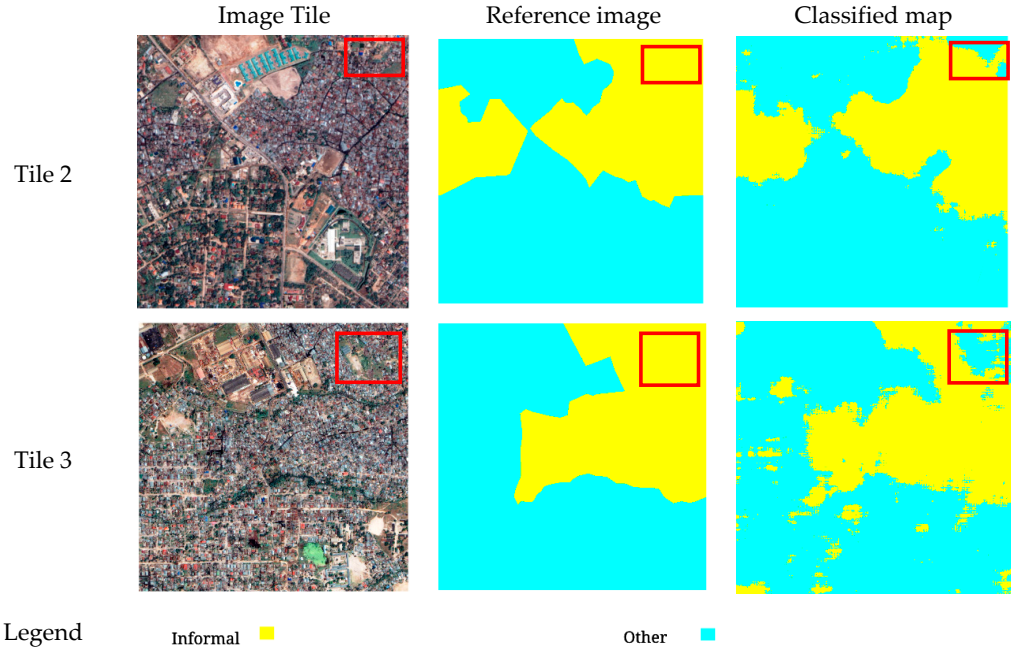

Figure 8. Red boxes highlight the northeastern part of tiles 1 and 2 where misclassification occur.

Table 5 reports user's accuracy (UA) and producer's accuracy (PA) [48] computed from the combined confusion matrices of tile 1, tile 2 and tile 3. We observe that CNN-5 has higher accuracy for both "informal" and "other" classes. In the "informal" class, CNN-5 has a PA of 91.40% and a UA of 88.22%, whereas SVM + GLCM has a PA of 90.44% and a UA of 75.63%, and SVM + LBP has a PA of 92.37% and a UA of 83.87%.

Training CNN from scratch using a patch size of $m = 165$ takes an average time of four hours when training, and spatial feature learning hyperparameters are kept constant using an NVDIA Quadro K2200 4GB GPU. The extraction of LBP and GLCM features takes an average time of two hours. While CNN is computationally intensive compared to SVM + GLCM or SVM + LBP, it enables the spatial-contextual features to be learned automatically in an end-to-end fashion.

Table 5. Accuracy assessment for the methods SVM, SVM + GLCM and CNN computed by combining the confusion matrix of Tile 1, Tile 2 and Tile 3.

Approach	Overall Accuracy (%)	Class	Accuracy (%)		Error (%)	
			User	Producer	Commission	Omission
SVM	68.84	Informal	40.61	71.60	59.39	28.40
		Other	88.68	68.00	11.32	32.00
SVM + GLCM	86.60	Informal	75.63	90.44	24.37	9.56
		Other	94.39	84.65	5.61	15.35
SVM + LBP	90.48	Informal	83.87	92.37	16.13	7.63
		Other	95.13	89.35	4.87	10.65
CNN-2	86.32	Informal	84.71	84.71	15.29	15.29
		Other	87.38	87.37	12.62	12.63
CNN-3	88.97	Informal	81.56	91.38	18.44	8.62
		Other	89.66	87.58	10.34	12.42

Table 5. *Cont.*

Approach	Overall Accuracy (%)	Class	Accuracy (%)		Error (%)	
			User	Producer	Commission	Omission
CNN-4	90.51	Informal	85.29	91.14	14.71	8.86
		Other	94.17	90.11	5.83	9.89
CNN-5	91.71	Informal	88.22	91.40	11.78	8.60
		Other	94.17	91.92	5.83	8.08
CNN-6	91.53	Informal	87.70	91.43	12.30	8.57
		Other	94.22	91.60	5.78	8.40

4.4. Visualization of Feature Maps

Figure 9 visualizes the feature maps extracted by the CNN to explain why the CNN performs better. The visualization demonstrates that the lower layers detect basic features such as edges, whereas the higher layers evolve to detect more complex features. The CNN feature maps indicate regions and patterns of the input image that produce activations in the network. A feature map is generated when a filter with learned weights is applied to the input image. Higher values in the feature maps correspond to regions of strong activations in the input image [49,50]. The displayed feature maps were generated by upsampling the actual feature maps, using bilinear resampling, to the original size of the input image to enhance the visualization process. We visualize the features from CNN-5 when tile 1 is used as an input. The rows indicate the eight feature maps that are generated after each of the convolutional layers (represented by the columns). In the 1st and 2nd layers, edges are more prominent (see Row 2, Column 1; Row 2, Column 2; Row 4, Column 1). In the higher layers, the feature maps are class specific and regions with informal settlements can be observed for example in (Row 3, Column 4; Row 3, Column 5; Row 5, Column 5). The lower layers of the CNN detect low level features while the higher layers detect more abstract features related to the semantic classes. Our experiments demonstrate that deep architecture can learn discriminative hierarchical spatial features. Intermediate layers distinguish the local information well, whereas the higher layers better discriminate the semantics classes. Indeed, in the highest layers, we observe that high values correspond to areas where the network detects the presence of informal settlements, while low activations corresponds to the class "other".

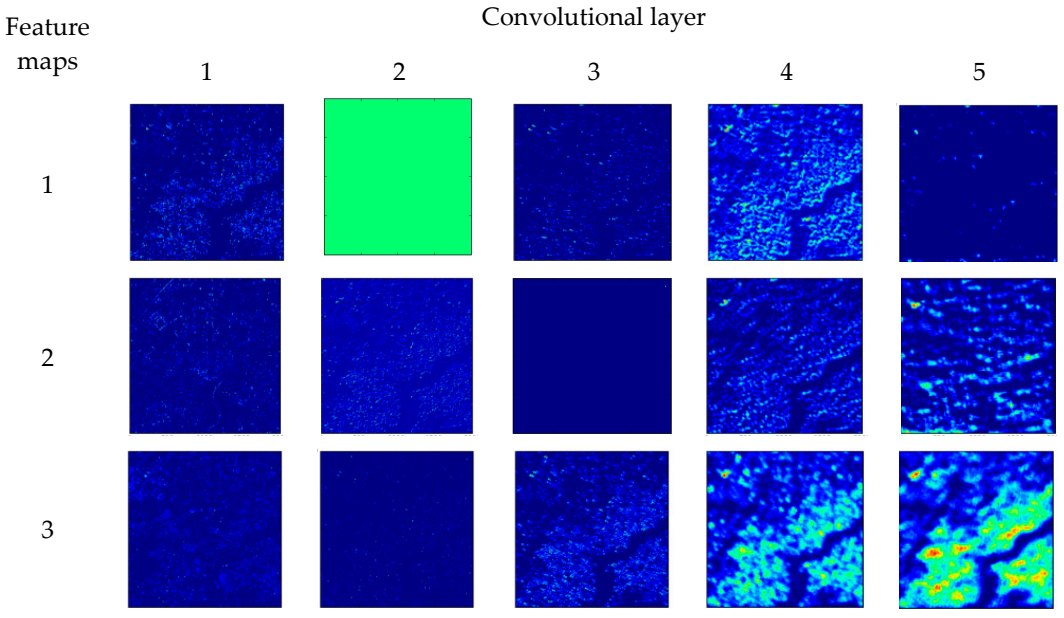

Figure 9. *Cont.*

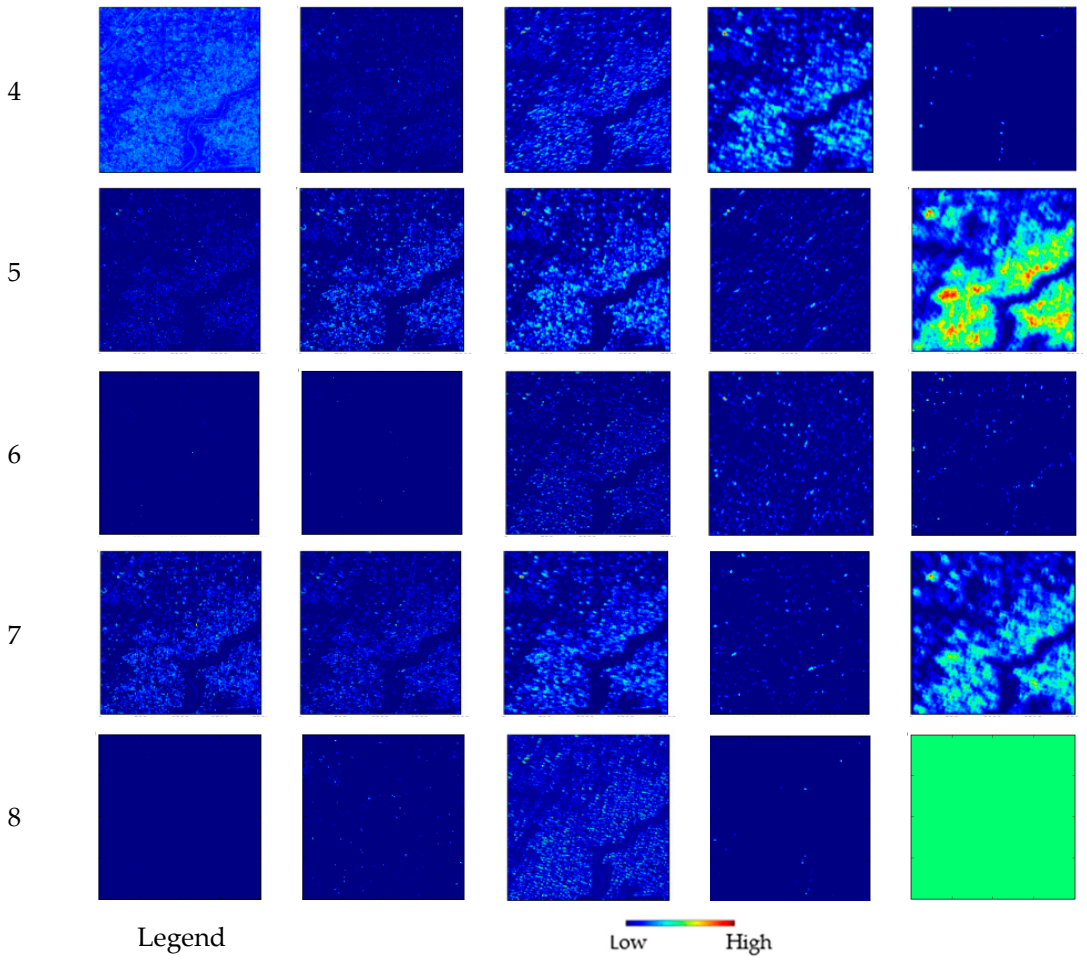

Figure 9. An illustration of eight feature maps for tile 1, derived from a CNN with five layers for each of the layers. The feature maps are upsampled through bilinear interpolation to attain a resolution of 2000 × 2000 pixels for visualization. High values are obtained where there is a strong response in the activations as the filter convolves over the input image.

5. Discussion

In this research, we investigated the use of CNN for the detection of informal settlements. CNN hyperparameter optimization experiments show that the depth of the network and the patch size have a significant influence on the network accuracy. A deep network allows for a hierarchy of useful spatial-contextual features to be learnt [51]. Deep networks are able to effectively look into a larger area of the input image as opposed to shallow networks, because the effective size of the receptive field of successive convolutional layers increases as the number of convolutional layers increase [31]. A large patch size allowed the extraction of spatial-contextual information from a large window. The optimal values of the CNN hyperparameters is guided by the particular classification task. We observed the importance of the training set size. Using a large training set is necessary to learn complex features and obtaining high generalization ability. In this work, training samples are derived from three tiles that contain informal settlements with different characteristics. Accuracy assessment across the three tiles showed that CNN is promising with regard to generalisation in the detection of informal settlements. Presently, urban settlements have a heterogeneous appearance based on their geographical location [12]. It would be desirable to have a method that is transferrable to detect informal settlements regardless of their location around cities in rapidly developing countries.

Visually, the quality of maps classified by CNN appear quite regular as shown in Figure 7. Nonetheless, the presence of islands in the classified maps could have been smoothened by using

simple post-processing. Here, we do not perform post-classification as we aimed to use CNN in an end-to-end fashion, from raw pixels to desired classes. We were able to visualize our CNN in order to gain an insight of the learned features. While most neurons were able to specialise, there were examples of some that failed to specialise as seen in (Row 1, Column 2; Row 8, Column 5) in Figure 9. This is due to the limitation of those RELU units which have zero gradient when inactive, which makes it impossible to adjust their weights during training [52]. Visualizing a trained CNN is also important because it can give insight into its design, for example, on the choice of the number of convolutional layers or dimension of kernel [50].

SVM + GLCM provided competitive results and the utility of GLCM features for the detection of informal settlements evident as shown in [12]. Similarly, LBP features are also useful for the detection of informal settlements as described in [14]. CNN showed the capability to learn spatial-contextual features directly from the input image, rather than hand-engineering. The high classification accuracy and visual quality of the maps demonstrate that CNNs are useful for the detection of informal settlements from VHR imagery.

Some uncertainties and inaccuracies in the classified maps were shown in Figure 8. The difficulty in correctly classifying these areas can be a result of an area having morphological characteristics of formal settlements, yet occurring within an informal settlement. Some work on evaluating uncertainty in image interpretation of informal settlements has been done in [53]. Nonetheless, the quantification of the magnitude and nature of these uncertainties could be evaluated in future studies.

Classified land use maps indicate the location and extent of informal settlements in a study area. This information is necessary to support decision making, planning and management of urban areas [54]. For instance, maps of informal settlements could be used to guide government and municipalities to properly allocate funding and resources for upgrading projects [55]. The upgrading process aims to improve social, organisational and environmental aspects of an informal settlement [56]. Maps can be used for planning specific interventions such as building new roads and infrastructures for access to clean water and sanitization [57,58]. In some situations, maps could be used to decide whether a slum should be eradicated and people moved to a more appropriate part of the city [59]. In addition, the resulting land use maps are useful in evaluating the sustainability in land use, land cover changes and transition when considered over a time series [60,61].

6. Conclusions

Our experiments showed that classification of VHR images using spectral features alone has a low accuracy and poor quality of maps. When spatial-contextual features are added, a high classification accuracy is obtained. SVM relying on GLCM features and LBP features performs well in this regard, hence well-designed hand-crafted features can exhibit competitive performance in the presence of classes with a high level of semantic abstraction. Our CNN had a high classification accuracy and outperformed SVM + GLCM and SVM + LBP, especially if a higher number of convolutional layers and a large training set were used. A deeper network allowed more discriminative spatial-contextual features to be learned, which help in separating complex classes. CNNs require an adequate training set to ensure the optimal determination of the parameters of the network. The methodology outlined in this paper can be replicated to map informal settlements in other cities. We conclude that CNNs, trained in an end-to-end fashion, can effectively learn complex, hierarchical and abstract features for a complex land use classification task, such as detection of informal settlements from VHR images. Future work may involve investigating transferability of this method for the detection of informal settlements in other geographical regions.

Acknowledgments: We acknowledge Monica Kuffer, Richard Sliuzas for providing the QuickBird dataset.

Author Contributions: Nicholus Mboga wrote the manuscript, designed and conducted the experiments. Claudio Persello, Alfred Stein and John Ray Bergado contributed to the conceptual design of the experiments, reviewed and revised the paper.

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

References

1. Kironde, J.M.L. The regulatory framework, unplanned development and urban poverty: Findings from Dar es Salaam, Tanzania. *Land Use Policy* **2006**, *23*, 460–472. [CrossRef]
2. UN-Habitat. *The Challenge of Slums—Global Report on Human Settlements*; UN-Habitat: Nairobi, Kenya, 2003.
3. UN-Habitat. Habitat. Habitat III Issue Papers 22—Informal Settlements. In *United Nations Conference on Housing and Sustainable Urban Development*; UN-Habitat: Nairobi, Kenya, 2015; Volume 2015, pp. 1–8.
4. Owen, K.K.; Wong, D.W. An approach to differentiate informal settlements using spectral, texture, geomorphology and road accessibility metrics. *Appl. Geogr.* **2013**, *38*, 107–118. [CrossRef]
5. Kuffer, M.; Barros, J.; Sliuzas, R.V. The development of a morphological unplanned settlement index using very-high-resolution (VHR) imagery. *Comput. Environ. Urban Syst.* **2014**, *48*, 138–152. [CrossRef]
6. Hofmann, P.; Strobl, J.; Blaschke, T.; Kux, H. Detecting informal settlements from Quickbird data in Rio de Janeiro using an object based approach. *Object-Based Image Anal.* **2008**, 531–553. [CrossRef]
7. Kuffer, M.; Barros, J. Urban Morphology of Unplanned Settlements: The Use of Spatial Metrics in VHR Remotely Sensed Images. *Procedia Environ. Sci.* **2011**, *7*, 152–157. [CrossRef]
8. Shekhar, S. Detecting slums from Quickbird data in Pune using an object oriented approach. *ISPRS—Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2012**, *XXXIX-B8*, 519–524. [CrossRef]
9. Bergado, J.R.A.; Persello, C.; Gevaert, C. A deep learning approach to the classification of sub-decimeter resolution aerial images. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium, Beijing, China, 10–15 July 2016; pp. 1516–1519.
10. Haralick, R.; Shanmugan, K.; Dinstein, I. Textural features for image classification. *IEEE Trans. Syst. Man Cybern.* **1973**, *3*, 610–621. [CrossRef]
11. Pesaresi, M.; Benediktsson, J.A. A New Approach for the Morphological Segmentation of High-Resolution Satellite Imagery. *IEEE Trans. Geosci. Remote Sens.* **2001**, *39*, 309–320. [CrossRef]
12. Kuffer, M.; Pfeffer, K.; Sliuzas, R.; Baud, I. Extraction of Slum Areas From VHR Imagery Using GLCM Variance. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2016**, *9*, 1830–1840. [CrossRef]
13. Kit, O.; Lüdeke, M.; Reckien, D. Texture-based identification of urban slums in Hyderabad, India using remote sensing data. *Appl. Geogr.* **2012**, *32*, 660–667. [CrossRef]
14. Ella, L.P.A.; van den Bergh, F.; van Wyk, B.J.; van Wyk, M.A. A Comparison of Texture Feature Algorithms for Urban Settlement Classification. In Proceedings of the 2008 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Boston, MA, USA, 6–11 July 2008; Volume 3, pp. III-1308–III-1311.
15. Graesser, J.; Cheriyadat, A.; Vatsavai, R.R.; Chandola, V.; Long, J.; Bright, E. Image based characterization of formal and informal neighborhoods in an urban landscape. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2012**, *5*, 1164–1176. [CrossRef]
16. Fauvel, M.; Tarabalka, Y.; Benediktsson, J.A.; Chanussot, J.; Tilton, J.C. Advances in Spectral—Spatial Classification of Hyperspectral Images. *Proc. IEEE* **2013**, *101*, 652–675. [CrossRef]
17. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
18. Schmidhuber, J. Deep Learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
19. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*; Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2012; pp. 1097–1105.
20. Ciresan, D.; Meier, U.; Schmidhuber, J. Multi-column Deep Neural Networks for Image Classification. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 3642–3649.
21. Paisitkriangkrai, S.; Sherrah, J.; Janney, P.; van den Hengel, A. Semantic Labeling of Aerial and Satellite Imagery. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2016**, *9*, 2868–2881. [CrossRef]
22. Längkvist, M.; Kiselev, A.; Alirezaie, M.; Loutfi, A. Classification and Segmentation of Satellite Orthoimagery Using Convolutional Neural Networks. *Remote Sens.* **2016**, *8*, 329. [CrossRef]
23. Alshehhi, R.; Reddy, P.; Lee, W.; Dalla, M. Simultaneous extraction of roads and buildings in remote sensing imagery with convolutional neural networks. *ISPRS J. Photogramm. Remote Sens.* **2017**, *130*, 139–149. [CrossRef]

24. Kussul, N.; Lavreniuk, M.; Skakun, S.; Shelestov, A. Deep Learning Classification of Land Cover and Crop Types Using Remote Sensing Data. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 778–782. [CrossRef]
25. Mnih, V. Machine Learning for Aerial Image Labeling. Ph.D. Thesis, University of Toronto, Toronto, ON, Canada, 2013.
26. Luus, F.P.S.; Salmon, B.P.; Van Den Bergh, F.; Maharaj, B.T.J. Multiview Deep Learning for Land-Use Classification. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 2448–2452. [CrossRef]
27. Castelluccio, M.; Poggi, G.; Sansone, C.; Verdoliva, L. Land Use Classification in Remote Sensing Images by Convolutional Neural Networks. *arXiv* **2015**; 1–11, arXiv:1508.00092.
28. Ševo, I.; Avramović, A. Convolutional Neural Network Based Automatic Object Detection on Aerial Images. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 740–744. [CrossRef]
29. Marmanis, D.; Datcu, M.; Esch, T.; Stilla, U. Deep Learning Earth Observation Classification Using ImageNet Pretrained Networks. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 105–109. [CrossRef]
30. Volpi, M.; Tuia, D. Dense semantic labeling of sub-decimeter resolution images with convolutional neural networks. *IEEE Geosci. Remote Sens. Lett.* **2017**, *55*, 881–893. [CrossRef]
31. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; The MIT Press: Cambridge, MA, USA, 2016.
32. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS), Sardinia, Italy, 13–15 May 2010; Volume 9, pp. 249–256.
33. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–25 June 2010; pp. 807–814.
34. Kivinen, J.J.; Williams, C.K.I. Transformation equivariant Boltzmann machines. In *Lecture Notes Computer Science(LNCS) (Including Its Subseries Lecture Notes Artificial Intelligence and Lecture Notes in Bioinformatics)*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6791, pp. 1–9.
35. Zeiler, M.D.; Fergus, R. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv* **2013**, arXiv:1301.3557.
36. Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*; Springer: Cham, Swizerland, 2010; pp. 177–186.
37. LeCun, Y.A.; Bottou, L.; Orr, G.B.; Müller, K.-R. Efficient BackProp. In *Neural Networks: Tricks of the Trade*, 2nd ed.; Montavon, G., Orr, G.B., Müller, K.-R., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 9–48.
38. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
39. Sliuzas, R.V.; Hill, A.; Lindner, C.; Greiving, S. *Dar es Salaam Land Use and Informal Settlement Data Set*; NASA Socioe: Palisades, NY, USA, 2016.
40. Sliuzas, R.V. Managing Informal Settlements: A Study Using Geo-Information in Dar es Salaam, Tanzania. Ph.D. Dissertation, Utrecht University, Utrecht, The Netherlands, 2004.
41. Theano Development Team. Theano: A {Python} framework for fast computation of mathematical expressions. *arXiv* **2016**, arXiv:1605.02688.
42. R Core Team. *R: A Language and Environment for Statistical Computing*. Available online: https://www.r-project.org/ (accessed on 23 October 2017).
43. Rossum, G. *Python Reference Manual*; CWI (Centre for Mathematics and Computer Science): Amsterdam, The Netherlands, 1995.
44. Farabet, C.; Couprie, C.; Najman, L.; Lecun, Y. Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1915–1929. [CrossRef] [PubMed]
45. Bruzzone, L.; Persello, C. Handbook of Pattern Recognition and Computer Vision. In *Handbook of Pattern Recognition and Computer Vision*; Chen, C.H., Ed.; World Scientific: Singapore, 2010; pp. 329–352.
46. Ojala, T.; Pietikäinen, M.; Mäenpää, T. A generalized local binary pattern operator for multiresolution gray scale and rotation invariant texture classification. *Adv. Pattern Recognit.* **2001**, *2013*, 399–408.
47. Ojala, T.; Pietikäinen, M.; Mäenpää, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 971–987. [CrossRef]
48. Congalton, R.G. A review of assessing the accuracy of classifications of remotely sensed data. *Remote Sens. Environ.* **1991**, *37*, 35–46. [CrossRef]
49. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef] [PubMed]

50. Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks. In *Computer Vision–ECCV 2014*; Springer: Cham, Swizerland, 2014; Volume 8689, pp. 818–833.
51. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* **2014**.
52. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In Proceedings of the 30th International Conference on Machine Learning, Atlanta, GA, USA, 16–21 June 2013; Volume 28, p. 6.
53. Kohli, D.; Stein, A.; Sliuzas, R. Uncertainty analysis for image interpretations of urban slums. *Comput. Environ. Urban Syst.* **2016**, *60*, 37–49. [CrossRef]
54. Marconcini, M.; Esch, T.; Chrysoulakis, N.; Düzgün, H.S.; Tal, A.; Forth, T.H.; Aviv, T. Towards EO-based sustainable urban planning and management. In Proceedings of the 2013 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Melbourne, Australia, 21–26 July 2013; pp. 4273–4276.
55. Sliuzas, R. Opportunities for enhancing communication in settlement upgrading with geographic information technology-based support tools. *Habitat Int.* **2003**, *27*, 613–628. [CrossRef]
56. Ward, P.M. Housing rehab for consolidated informal settlements: A new policy agenda for 2016 UN-Habitat III. *Habitat Int.* **2016**, *50*, 373–384. [CrossRef]
57. Hasan, A. Orangi Pilot Project: The expansion of work beyond Orangi and the mapping of informal settlements. *Environ. Urban.* **2006**, *18*, 451–480. [CrossRef]
58. Karanja, I. An enumeration and mapping of informal settlements in Kisumu, Kenya, implemented by their inhabitants. *Environ. Urban.* **2010**, *22*, 217–239. [CrossRef]
59. Abbott, J. An analysis of informal settlement upgrading and critique of existing methodological approaches. *Habitat Int.* **2002**, *26*, 303–315. [CrossRef]
60. Marais, L.; Ntema, J. The upgrading of an informal settlement in South Africa: Two decades onwards. *Habitat Int.* **2013**, *39*, 85–95. [CrossRef]
61. Musakwa, W.; Niekerk, A. Van Implications of land use change for the sustainability of urban areas: A case study of Stellenbosch, South Africa. *Cities* **2013**, *32*, 143–156. [CrossRef]