Journal of
Applied Remote Sensing

RemoteSensing.SPIEDigitalLibrary.org

Semantic segmentation on small datasets of satellite images using convolutional neural networks

Mohammed Chachan Younis
Edward Keedwell

SPIE.

Mohammed Chachan Younis, Edward Keedwell, "Semantic segmentation on small datasets of satellite images using convolutional neural networks," *J. Appl. Remote Sens.* **13**(4), 046510 (2019), doi: 10.1117/1.JRS.13.046510.

Semantic segmentation on small datasets of satellite images using convolutional neural networks

Mohammed Chachan Younis* **and Edward Keedwell**
University of Exeter, College of Engineering, Mathematics, and Physical Sciences, Computer Science Department, Exeter, United Kingdom

Abstract. Semantic segmentation is one of the most popular and challenging applications of deep learning. It refers to the process of dividing a digital image into semantically homogeneous areas with similar properties. We employ the use of deep learning techniques to perform semantic segmentation on high-resolution satellite images representing urban scenes to identify roads, vegetation, and buildings. A SegNet-based neural network with an encoder–decoder architecture is employed. Despite the small size of the dataset, the results are promising. We show that the network is able to accurately distinguish between these groups for different test images, when using a network with four convolutional layers. © *2019 Society of Photo-Optical Instrumentation Engineers (SPIE)* [DOI: 10.1117/1.JRS.13.046510]

Keywords: deep learning; convolutional neural networks; semantic segmentation; satellite images.

Paper 190501 received Jul. 2, 2019; accepted for publication Oct. 25, 2019; published online Nov. 18, 2019.

1 Introduction

Deep learning has received growing interest in the last 10 years due to its unprecedented capability in the processing of images. Due to the availability of higher computational power and the versatility of neural networks, deep learning techniques have been applied in many fields of research, outperforming traditional machine learning methodologies. Deep neural networks are generic models that are able to model any multivariate nonlinear relationship, given a sufficient number of neurons and layers. For this reason, they can be employed for classification, regression, clustering, and generative processes, and they are able to process complex data such as digital signals (audio, images, and videos).[1,2]

A popular and promising application of deep learning is semantic segmentation. Semantic segmentation refers to the division of an image into semantically homogeneous areas,[3] which means that every pixel in a given area is associated with the same meaning (that is, the same in some sense). For example, an image representing an indoor scene could include a chair, table, person, and background, whereas an image representing an outdoor scene could include mountains, fields, beaches, roads, and buildings. Thus, semantic segmentation is a particular case of classification where each pixel of the image is classified according to the probability of each class.

Convolutional neural networks (CNNs) have been proven to be effective in image segmentation.[4,5] Indeed, the most popular algorithms for semantic segmentation employ CNN as this is the most suitable architecture to process images since it is very efficient and effective,[6] and it can even be employed for real-time applications.[7]

More generally, CNN are particularly suited for image processing. The most important feature of CNN is the convolutional layer. This layer convolves the input with a certain number of filters. Each filter is able to capture a specific feature (e.g., edges and corners),[8] and each time that feature is detected in the image, the filter outputs an increased value. The outputs of the filters are aggregated to form a new representation of the input, and the more convolutional layers there are, the more abstract and complex the representation is.[9] The first layers are able to capture basic geometric features, while higher levels may model features with high-level semantics and

*Address all correspondence to Mohammed Chachan Younis, E-mail: mcy201@exeter.ac.uk

complex shapes (e.g., faces, cars, and trees). Convolutional layers are usually followed by a pooling layer with the purpose of reducing the dimensionality of the input[10,11] and nonlinear functions [sigmoid, rectified linear unit (ReLU), and hyperbolic tangent] to introduce the ability to model nonlinear relationships.

Semantic segmentation has been applied in different scenarios. These include, for example, urban scenes,[12] indoor scenes,[13] outdoor scenes,[14] and autonomous driving,[15] whereas several studies focus on satellite images.[16] The purpose of the CNN model in this application is to divide the image into the elements that characterize a map, such as vegetation, buildings, and roads, to provide a real-time application range from coverage mapping to urban planning. This task is particularly difficult since elements belonging to the same class may exhibit a large variation in terms of shape, color, and texture. Moreover, it is difficult to collect a large dataset for the training stage.

In the deep learning field, it is commonly known that a large amount of data is required to properly train a network. This concept gets stronger every year, as the trend in the AI community is to research always deeper and more complex networks. Unfortunately, accessing a suitable amount of data is not possible for everyone along with data ground truth information, thus making difficult to train a large network for a custom application.

In this paper, we introduce a methodology, implemented in MatLab R2018b, for semantic segmentation on RGB satellite images with a dataset of limited size. One of the goals of this work is to acquire a high-quality CNN using a small dataset. We consider three classes: buildings, vegetation, and roads. We employ a CNN with an encoder–decoder architecture based on SegNet. The data are processed with different augmentation techniques and the best network architecture is searched by running several experiments where the important parameters are tuned. We show that even with a small number of training images promising results can be achieved.

The choice of a CNN is motivated by the multitude of studies that prove the general superiority of this approach over traditional methods, as explained in the next section.

Six high-resolution satellite images are initially used and shown in Fig. 1. It is possible to see that they can be grouped into three groups in terms of similarity (first and second; third and fourth; fifth and sixth). For this reason, we consider them separately in the training stage and in the results analysis.

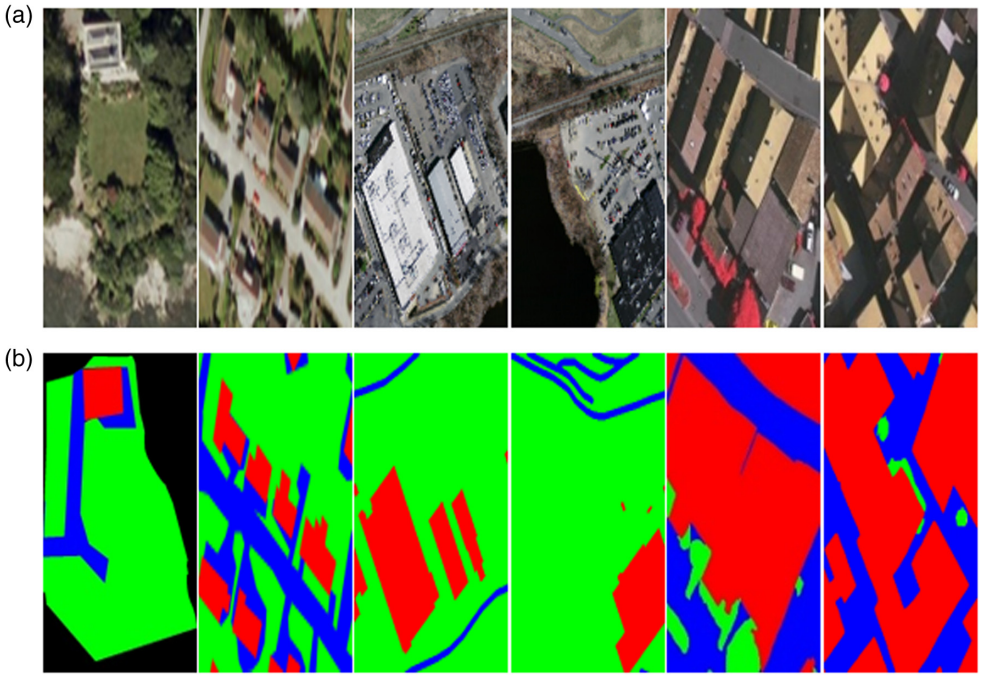

Fig. 1 (a) The six images that we considered for this study, together with (b) their labeled ground truth. (Images are taken from Ref. 17.)

To produce optimal results, the training dataset should have the following characteristics where possible: (1) class balance: every class should appear in the dataset with approximately the same frequency (same number of samples/observations). For example, images with 95% volume of vegetation class and just one small building class will result in poor classification performance on the small class. Theoretically, if some classes have a low probability, these will have a low accuracy of determination, because CNN net has poor training for this. See Ref. 18 for more details. (2) Intraclass homogeneity: pixels/areas belonging to the same class should be similar to each other. For example, if all the areas belonging to vegetation are green in the RGB image, red trees are unlikely to be correctly classified. Similarly, if the network learns that all buildings are rectangular, a building with another shape may be assigned to another class. See Ref. 19 for more details. (3) Scale: the images used should have the same approximate zoom level. Different sized images make it difficult to create a model. See Ref. 20 for more details. (4) Dataset size: the more images used for the better the results seen, particularly for CNNs. See Ref. 21 for more details.

2 Related Work

2.1 *Semantic Segmentation*

Semantic segmentation was addressed before the advent of deep learning, with popular algorithms such as watershed segmentation,[22,23] semantic texton (the elements of texture perception) forests,[24] and random forest-based classifiers.[25]

In satellite image segmentation, several approaches have been tried. In Ref. 26, two swarm-intelligence-based global optimization algorithms for multilevel thresholding were employed, obtaining good results for satellite image segmentation. Bhandari et al.[27] presented a more computationally efficient algorithm, in terms of accuracy and computational time, for satellite image segmentation based on a modified artificial bee colony.

2.2 *Convolutional Networks*

The advent of the neural network has had a considerable impact on image processing. CNNs show excellent performance with respect to state-of-the-art methods both for semantic segmentation and other applications. Generally speaking, it can be said that deep learning-based methods outperform the traditional ones.[28]

In 2014, fully convolutional networks[4] were shown to be able to produce dense predictions without any fully connected layers, allowing much faster predictions for large images. The subsequent works on deep learning-based semantic segmentation followed this paradigm.

In 2015, SegNet was introduced.[7] SegNet is a fully deep convolutional network designed for image segmentation. It is based on an encoder–decoder architecture, with a high number of convolutional layers. There are no fully connected layers, reducing the number of parameters of the network. The final layer produces a probability value for each pixel in the original image.

An important feature in SegNet is the use of maxpooling indices in the decoder to perform upsampling of low-resolution features. When maxpooling is performed in the encoder, the locations of the maximum feature value in each pooling window are stored and used by the decoder. As a consequence, high-frequency details are retained in the segmented images, preventing blurred boundaries, and the total number of trainable parameters in the decoder is reduced. The architecture is trained end-to-end using stochastic gradient descent (SGD). The network is tested on several test cases, such as urban scenes and indoor scenes, obtaining impressive results.

Although it has not been designed specifically for satellite images, we believe that architecture similar to SegNet is particularly applicable to this domain. As we will explain later, we use this structure as a base reference and then subsequently reduce the number of layers.

2.3 *Other Work Related to This Problem*

The literature on semantic segmentation includes some works that face a problem similar to the one that is presented in this paper and are used as a baseline to compare our method, as shown in

Sec. 4. Here a short description of these works, which are mostly based on traditional methods, is provided.

Gamba and Houshmand[29] combined the multimodal data coming from remote sensors to model the shape of buildings and land cover. Fuzzy c-means clustering algorithms are employed. In Refs. 30 and 31, traditional classification methods based on decision trees are employed on aerial multispectral images. In Ref. 32, features are extracted from high-resolution aerial images and used to train pixel-based (support vector data description, Gaussian mixture model, and nearest-neighbor) and object-based classifiers (eCognition) of vegetation and urban areas. In Ref. 33, segmentation on nine categories from remotely sensed images using genetic sequential image segmentation, an iterative segmentation algorithm, tries to optimize the local balance between coverage, consistency, and smoothness of each class. In Ref. 34, a combination of low-computation algorithms is employed on aerial orthophotography and digital elevation model (DEM) data and a 7-class segmentation task. In Ref. 35, a knowledge-based system is used on multimodal data in order to better discriminate between asphalt road, vegetation, and nonvegetation.

3 Methodology

3.1 *Data Preparation*

Six RGB images (I_1, I_2, I_3, I_4, I_5, and I_6) of various sizes, representing a large-scale urban landscape, were used as the first training set. The ground truth was manually built by labeling the pixels according to three different classes: buildings, vegetation, and roads. These images were then splitted into 128×128 sized subimages.

Since this dataset is quite limited for a semantic segmentation task, several augmentation techniques were employed to make it larger. In particular, affine transformations, brightness transformation, and the addition of noise were used. These techniques are typically used in deep learning[36] with affine transformations, including horizontal and vertical flipping and rotation with a random angle.[37] Brightness transformation was also randomly applied to each image, while the noise used was Gaussian.[38]

Technically, the augmentation process transforms the training images in such a way that for the neural network they are different, increasing the diversity of the data, and preventing the network from memorizing the exact details of the existing images.[39]

For each individual function and for each of the three groups of functions, the probability of their occurrence is given, as well as the range of values that the function can accept. Further, a random number of functions that will take part in the transformations is randomly selected, then one is selected from the available range of values in the same way, after which the selected functions process the image in turn, with the results that all received effects overlap each other. This process increases the diversity of the data.

Each large image resulted in a number of subimages ranging from 212 to 1164, after augmentation. Examples of such image augmentations are shown in Figs. 2 and 3. Then the images were divided as follows.

We ran the training using a sixfold cross-validation strategy. At each training iteration, the subimages coming from five original images were used for training and validation, while the images coming from the remaining one were used for testing. In this way, all the images contributed to the training and testing without overlapping and, at the same time, we perform validation to assess the accuracy of the network and monitor the presence of overfitting.

Our dataset is summarized in Table 1.

An analysis stage was conducted where the occurrence of each class in the six images was checked. If the classes are not balanced in the dataset, some remedial action needs to be taken. Figure 4 shows the results of this analysis. It can be seen from this chart that the dataset is not balanced: vegetation has a much higher frequency of occurrence in the first 4 images, while for images 5 and 6 the number of pixels related to vegetation was much lower than the other classes. In every image, roads have a lower frequency with respect to the other classes. As we explain in the next section, we take this issue of dominant classes into account by means of class weights.

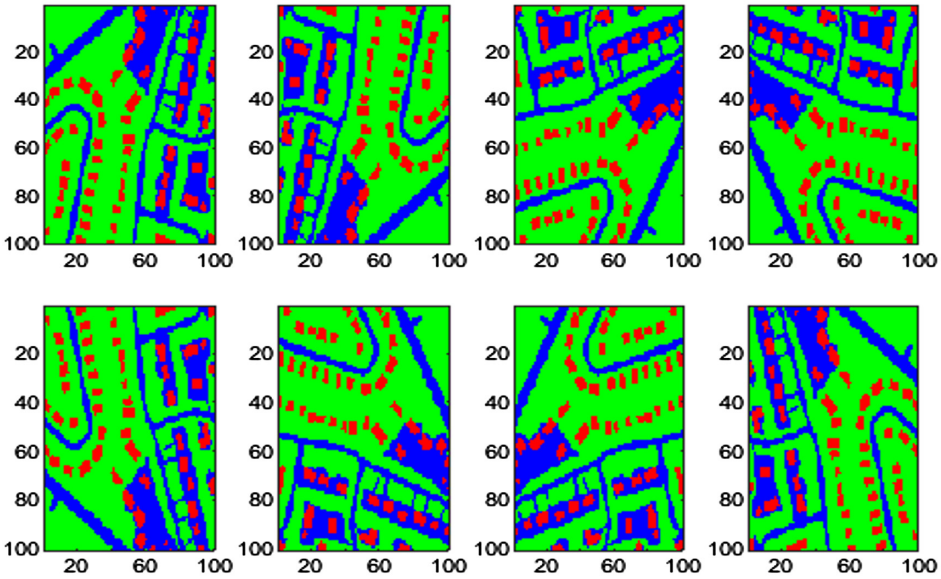

Fig. 2 Example of image augmentation. Original image (upper left) and augmented images, generated by rotation (90 deg, 180 deg, and 270 deg) and reflection (up/down, left/right).

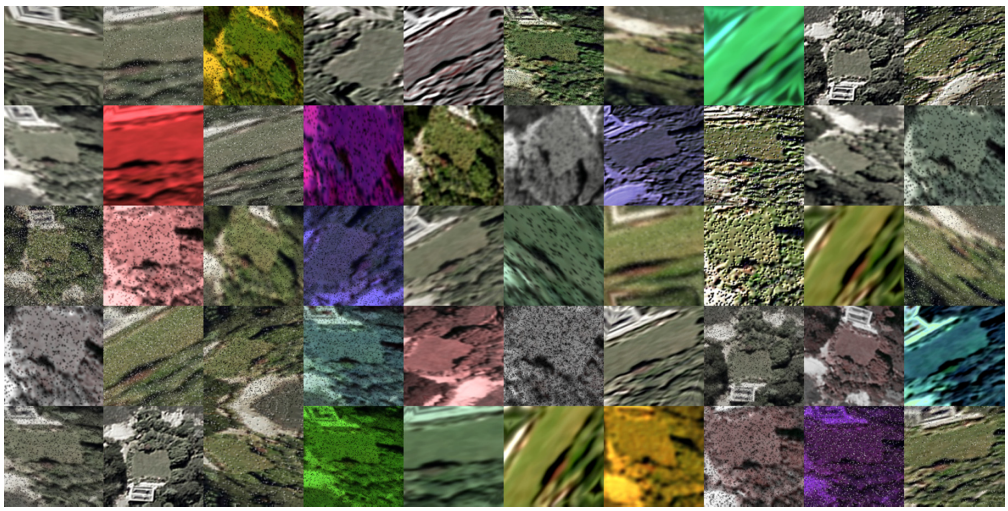

Fig. 3 Set of augmented images for image 1.

Table 1 Dataset details.

Original image	No. of subimages after augmentation
1	212
2	237
3	372
4	372
5	1002
6	1164
Total	3359

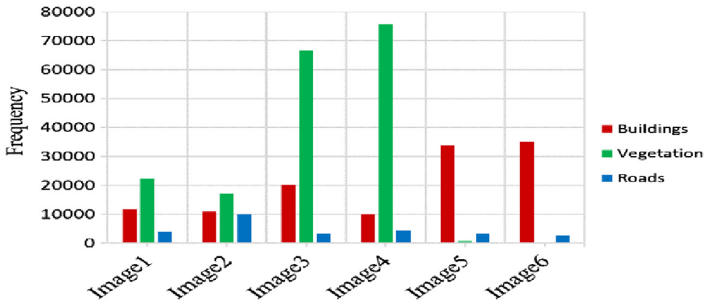

Fig. 4 Occurrence of each class in the six images.

3.2 *Data Preprocessing*

The preprocessing step is crucial in neural network training and must be carefully planned in order to make the learning faster and more stable. In particular, it is known that normalizing the input data to a fixed range produces better classification results.[40]

Each input image is resized to a fixed size ($128 \times 128 \times 3$) in order to have a good trade-off between too large images (long training time) and too small images (bad classification performance). Histogram equalization is performed on each RGB channel in order to increase the contrast and improve the network performance.[41] Then the images are normalized to the [0, 1] range. Normalization is typically performed in neural networks because the nonlinear functions that are employed work better in this range. Moreover, if the inputs have different ranges, with normalization, we bring them to the same range so that they are comparable.

3.3 *Network Architecture and Training*

For the network implementation, the architecture of SegNet was used as the starting point. The choice was motivated by the fact that this network achieves good results on different datasets and offers a structure that can be modified according to specific needs. SegNet is based on the encoder–decoder architecture. The encoder part takes an image as input and encodes it in a lower dimensional vector which contains the feature that best characterizes the image. It consists of several convolutional layers, each followed by batch normalization, ReLU nonlinearity, and a maxpooling. The dimensionality of the data is reduced after each pooling layer.

The decoder takes a vector of features as input and produces an image of the same size as the input. It reflects the same structure as the encoder, with an equal number of deconvolutional blocks followed by batch normalization, leaky ReLu, and upsampling. At the end, the SoftMax layer provides a probability value for each class.[42] Each pixel is assigned the class with the highest probability, since the purpose is to provide a classification which is as equal as possible to the ground truth.

The number of convolutional layers, the number of filters per layer, and the filter size are important parameters that determine the abstraction and modeling ability of a neural network. This number depends on the particular task that is being faced and has to be chosen carefully in order to avoid underfitting and overfitting. Moreover, the computational complexity and the memory requirement of the trained model depend on these parameters.

SegNet was conceived to be trained and tested on a large amount of data with many classes. For this reason, as typically found in the state-of-the-art deep learning, it employs a very high number of layers and has a particularly large dataset for the training phase. Since we do not have a large number of images at our disposal, as mentioned in Sec. 1, we modified the structure of the network. The number of convolutional layers and the number of parameters per layer was reduced, in order to prevent overfitting. The choice of these numbers was determined after a phase where different configurations were tested. The performance of the network was tested at each phase and the best configuration was chosen. We describe the different configurations in Sec. 4.1.

The architecture of our network is depicted in Fig. 5.

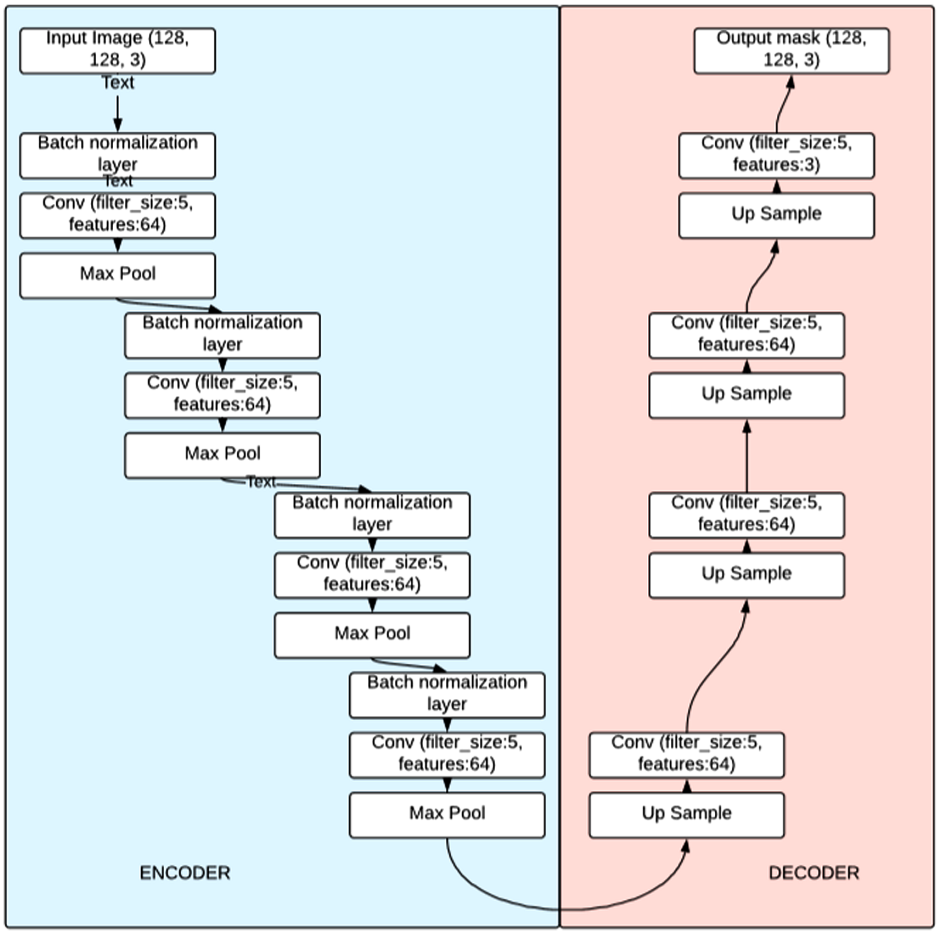

Fig. 5 Our network's architecture.

As the class distribution in our dataset was not balanced (the number of pixels related to vegetation was higher than the other classes), the SoftMax computation assigns a weight to each class,[43] based on the inverse of the frequency of occurrence (the rarer the class, the higher the weight). Weights are related to the probability of observing a given class. If all classes occur with equal frequency, there is no issue. But if a class is extremely rare, when the network is uncertain whether to predict that class or a more likely one, it will always predict the latter in order to have more probability to guess correctly. However, when using weights, Eq. (1), this problem is greatly reduced because it prevents the network from classifying every pixel to the most frequent class, which reduces classification error.

The weights are given by

$$w_i = \frac{\text{median}(p_i)}{p_i}, \tag{1}$$

where w_i is the weight associated to class i and p_i is the relative frequency of class i.[7]

3.4 *Training*

As mentioned above, we conducted the training phase using sixfold cross-validation. At each iteration, one of the six images was kept out and used for validation once the network was trained. The coefficients of the filters were initialized with a normal distribution. The network was trained using the SGD algorithm, with learning rate equal to 0.5, a drop factor of 0.5, and a drop period of 200 epochs (images are not fed one by one into the network in the training set, but are grouped in batches). The training algorithm uses cross entropy as the loss function

(see Ref. 44 for more details), which is commonly used in neural network-based image processing. Filters of various sizes were used (according to the network configuration) and stride 1 for the convolutions. The training accuracy was computed as the percentage of correctly classified pixels in the validation set. When the accuracy reached a stationary level, the training is stopped.

4 Results

In this section, we describe the results that were achieved for the test images, which are the six original images. Each test image was considered separately, showing the result of the prediction in terms of a segmented image, which offers an easier interpretation of the results through showing how accurately the image is predicted visually by comparing how well it matches with actual ground truth; and confusion matrices, which indicate the correct classification rate for each pair of classes by providing the vectors with predicted pixels and true pixels.

4.1 *Network Configurations and Training*

As introduced in the previous section, we conducted a comparison of the performance of different network configurations, starting from a simplified version of SegNet. The purpose was to find a good configuration for our limited dataset. Training and validation were performed in an iterative fashion. We considered three parameters: number of layers, number of filters per layer, and kernel size. At each iteration, different combinations of these parameters were chosen, and the training was performed. At the end, we compared the performance of the different networks that we trained. The comparison is illustrated in Table 1. The average training and validation accuracy achieved over the whole dataset were used as the performance metric. The configurations that achieved the worse results have been discarded.

From Table 2, it can be seen that the number of layers most strongly affects the accuracy (as clearly shown in Fig. 6). In terms of training and validation accuracy, the best model is a model with 4 deep layers and 64 feature maps. However, it is worth noting that the model with the maximum number of layers has a slightly lower accuracy, likely due to the attenuation of the gradient. It was also noted that the number of feature maps does not significantly affect the accuracy. Comparing models 10, 11, and 12, where the number of layers and feature maps is the same, but the filter size is different, we can say that the model with filter size 7 has the highest accuracy.

If too simple a network is used, the trained model is not able to correctly fit our data. For example, using just one layer, the validation accuracy does not exceed 62%. This is because only simple features (such as edges) have been captured. The highest validation accuracy (89%) is achieved using four layers. This appears to be one of the most important parameters as relevant changes were not seen when the filter size or the number of filters per layer was varied. Therefore, our chosen network configuration includes 4 layers with 64 filters per layer and a filter size equal to 5.

The training and validation plots are depicted in Fig. 7, relatively to the training stage with the dataset including images 1 to 5. For reasons of space, the plots related to the other cases are not displayed. However, the results were similar. In particular, two plots are displayed. The first is related to the accuracy in the training set, i.e., the percentage of correctly classified pixels (Fig. 7 green curve). An increasing accuracy means that the network is improving its prediction capability. Conversely, the second plot refers to the training loss/error measure (Fig. 7 red curve). The lower the loss, the higher the performance. The slope of these curves depends on the learning rate and on the state of the network. A higher learning rate means that the weights change faster, and so do the accuracy and the loss. At the beginning, we did not know whether the optimization algorithm reached a global minimum or a local minimum.[45] In the latter case, we needed a high change in the loss to proceed from the local minimum toward a better minimum. If the curve flattened, we could say that a local or global minimum had been reached, and when such a minimum was reached, each change in the weights did not affect the accuracy and loss significantly.

Table 2 Impact of different network configurations on results, where *K* is the size of the convolutional filters and L*i* is the *i*'th layer.

		No. of filters					Accuracy (%)	
No. of layers	K	L1	L2	L3	L4	L5	Train	Val
1	5	32	—	—	—	—	63.39	61.69
1	5	64	—	—	—	—	63.86	62.35
1	5	96	—	—	—	—	63.77	62.13
1	5	128	—	—	—	—	64.06	62.56
2	5	64	32	—	—	—	75.72	74.05
2	5	64	64	—	—	—	77.71	76.78
2	5	96	96	—	—	—	79.61	77.70
2	5	128	128	—	—	—	76.81	75.34
3	5	64	32	16	—	—	84.80	83.73
3	5	64	64	64	—	—	87.44	86.23
3	3	64	64	64	—	—	78.44	77.74
3	7	64	64	64	—	—	90.05	88.88
3	5	96	96	96	—	—	88.70	87.33
3	5	64	96	128	—	—	85.62	84.43
4	**5**	**64**	**64**	**64**	**64**	—	**90.73**	**89.24**
5	5	64	64	64	64	64	90.56	88.85

Note: The bold values indicate the best model and the corresponding train and validation accuracy.

The training was stopped when the accuracy reached a stationary value (about 200 epochs), meaning that further iterations would have produced no significant change in the network's weights.

As expected, and mentioned above, the validation accuracy is slightly lower than the training accuracy. The training has been stopped when the performance stopped improving, and the accuracy reached an almost constant value.

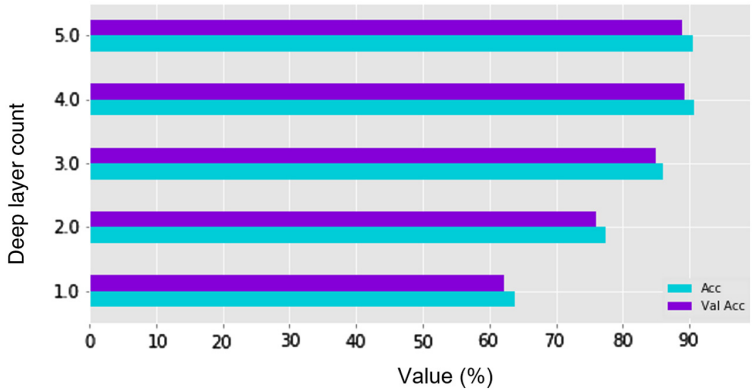

Fig. 6 Effect of the number of layers on accuracy. Acc is the training accuracy (%) and Val Acc is the validation accuracy (%).

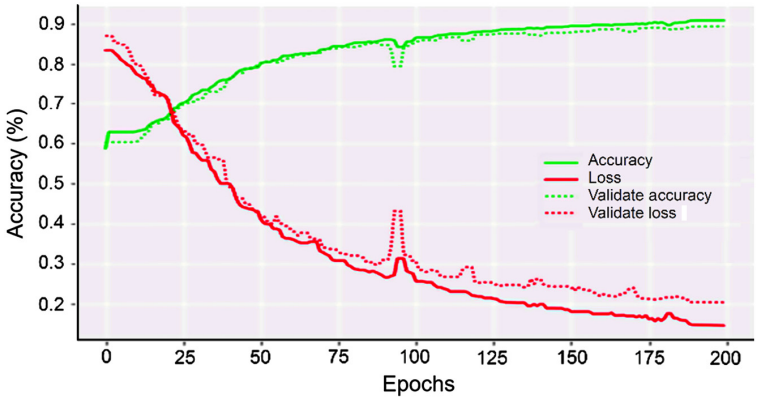

Fig. 7 Training accuracy and loss with the dataset for the chosen network architecture.

4.2 *Test Results*

In this section, the prediction results are depicted that were achieved in the six different test stages. As already mentioned, each time a different image was used as a test case.

By looking at the results, it is possible to draw some interesting observations. The vegetation has been well modeled by the network, even when the color is not green. For example, images 5 and 6 include red vegetation which is correctly segmented. The network is able to segment the roads, which, however, are not always segmented with straight edges. See, for example, test image 2. Moreover, it is possible to notice some confusion between roads and trees (image 1). The buildings are very well modeled, at different zoom levels.

In Table 3, standard metrics, precision, recall, F-score, kappa, and overall accuracy (OA) are presented over the different test stages. From this, it can be seen that the network is particularly good at predicting buildings with an OA of 93.67% and vegetation with an OA of 95.83%. As for the roads, the OA is lower (67.71). This can be explained by the scarcity of pixels related to roads in the datasets, as clearly shown in Fig. 4. We believe that the same architecture could perform much better even on roads, with a larger dataset. The low precision on roads, together with the accuracy values, indicates that many pixels that the network tends to classify is part of the roads as buildings or vegetation.

Precision and recall are combined in the F-score as shown in Eq. (2), a measure of test's accuracy which is given by

$$2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{2}$$

Based on the F-score, the class that is best modeled by the system is buildings. The relation between the predictions of the various classes is shown in detail in the confusion matrices and segmented images presented in Figs. 8–10. B, V, and R mean buildings, vegetation, and roads, respectively.

Table 3 Evaluation metrics.

	Buildings (%)	Vegetation (%)	Roads (%)
Producer accuracy (precision)	96.35	92.93	73.56
User accuracy (recall)	93.67	95.83	67.71
F-score	94.99	94.36	70.51
Kappa		86.7%	
Mean OA		92.63%	

(a)

Image 1		Actual values			
		B	V	R	Precision
Predicted values	B	11251	1032	295	89.4%
		29.6%	2.7%	0.8%	10.6%
	V	34	19123	1435	92.9%
		0.1%	50.3	3.8%	7.1%
	R	149	3351	1355	27.9%
		0.4%	8.8%	3.6%	72.1%
	Recall	98.4%	81.4	43.9	**83.4%**
		1.6%	18.6	56.1	16.6%

(b)

Image 2		Actual values			
		B	V	R	Precision
Predicted values	B	10048	412	440	92.2%
		26.4	1.1%	1.2%	7.8%
	V	622	15072	1483	87.7%
		1.6%	39.6%	3.9%	12.3%
	R	716	1154	8078	81.2%
		1.9%	3.0%	21.2%	18.8%
	Recall	88.2	90.6%	80.8%	**87.3%**
		11.8	9.4%	19.2%	12.7%

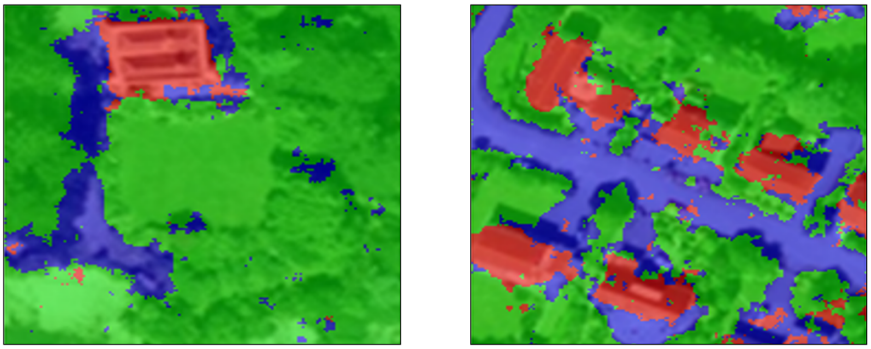

Fig. 8 Confusion matrices and predicted images for test (a) image 1 and (b) image 2.

(a)

Image 3		Actual values			
		B	V	R	Precision
Predicted values	B	17296	2933	4	85.5%
		19.2%	3.3%	0.0%	14.5%
	V	1537	64351	587	96.8%
		1.7%	71.5%	0.7%	3.2%
	R	60	1574	1658	50.4%
		0.1%	1.7%	1.8%	49.6%
	Recall	91.5%	93.5%	73.7%	**92.6%**
		8.5%	6.5%	26.3%	7.4%

(b)

Image 4		Actual values			
		B	V	R	Precision
Predicted values	B	8576	1371	66	85.6%
		9.5%	1.5%	0.1%	14.5%
	V	149	74224	1222	98.1%
		0.3%	82.5%	1.4%	1.9%
	R	7	1125	3160	73.6%
		0.0%	1.3%	3.5%	26.4%
	Recall	97.1%	96.7%	71.0%	**95.5%**
		2.9%	3.3%	29.0%	4.5%

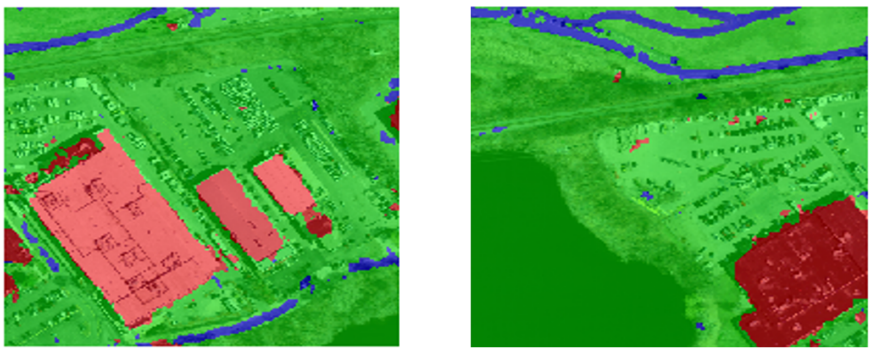

Fig. 9 Confusion matrices and predicted images for test (a) image 3 and (b) image 4.

(a)

Image 5		Actual values			
		B	V	R	Precision
Predicted values	B	33496	38	349	98.9%
		88.1%	0.1%	0.9%	1.1%
	V	59	626	120	77.8%
		0.2%	1.6%	0.3%	22.2%
	R	509	152	2676	80.2%
		1.3%	0.4%	7.0%	19.8%
	Recall	98.3%	76.7%	85.1%	**96.8%**
		1.7%	23.3%	14.9%	3.2%

(b)

Image 6		Actual values			
		B	V	R	Precision
Predicted values	B	34226	31	789	97.7%
		90.0%	0.1%	2.1%	2.3%
	V	82	143	115	42.1%
		0.2%	0.4%	0.3%	57.9%
	R	330	30	2279	86.4%
		0.9%	0.1%	6.0%	13.6%
	Recall	98.3%	70.1%	71.6%	**96.4%**
		1.2%	29.9%	28.4%	3.6%

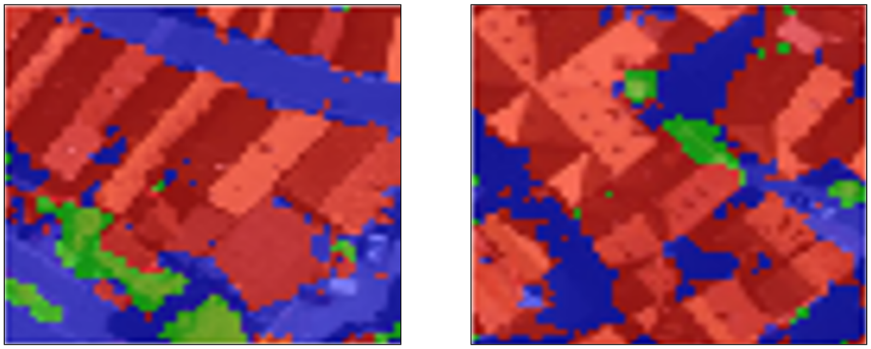

Fig. 10 Confusion matrices and predicted images for test (a) image 5 and (b) image 6.

Table 4 Strength of agreement for categorical data of kappa interpretation.

Kappa statistic	Interpretation
<0.00	Poor agreement
0.00 to 0.20	Slight agreement
0.21 to 0.40	Fair agreement
0.41 to 0.60	Moderate agreement
0.61 to 0.80	Substantial agreement
0.81 to 1.00	Almost perfect agreement

When dealing with an imbalanced dataset, it is essential to pay attention not only to the overall evaluation metrics but also the corresponding misclassification costs. Thus, kappa statistics are a good performance measure when facing an imbalanced dataset. Längkvist et al.[46] proposed a qualitative interpretation of kappa statistics (Table 4) which was assigned to the corresponding agreement measures.

4.2.1 *Images 1 and 2*

Here, the prediction results for images 1 and 2 are shown considering the network with the chosen configuration. The confusion matrices and segmented images are shown in Fig. 8. This concerns the network with four layers, so it can be seen that the performance of image 1 [Fig. 8(a)] is high for classes one and two, while it is very low for class three, suggesting that more abstraction and complexity is needed to model this class. Figure 8(b) shows the confusion matrix and the prediction for image 2. In this case, the prediction accuracy for the third

class is much higher. As can be seen in the predicted image, the content related to the third class is much clearer than the previous image, where the roads were covered by trees. This implies the necessity to introduce some more prediction ability to model hidden areas. This can be given by a larger dataset and a more complex network.

4.2.2 *Images 3 and 4*

The third image achieved good performance, although it is very different from images 1 and 2 in terms of content and class distribution. The OA is 92.6%, as shown in Fig. 9(a). Even better is the accuracy of image 4, which is shown in Fig. 9(b). The overall value is 95.5%. The roads are more difficult to distinguish with respect to image 3, producing a lower class-specific accuracy. This big influence on the accuracy of roads is owing to the fact that many regions around car parks, which all have the same color features as roads, are not marked as roads on the ground truth of both images. The images 3 and 4 have large car parks with cars in them, and look-like building roofs but are attached to the vegetation class, which raises a segmentation error.

4.2.3 *Images 5 and 6*

Images 5 and 6 are the ones that achieved the best performance, especially in terms of buildings and roads. As we can see in Fig. 10, the roofs were clearly discernible, and the network could segment them correctly. The confusion matrices indicate an accuracy of more than 97% for class one and more than 80% for class three, while the performance for class two is lower. This could be due to the shortage of vegetation in the training set for images 5 and 6.

4.3 *Comparison with Other Works*

In this section, the results that were achieved are compared with the works introduced in Sec. 2. Although the datasets are not the same (e.g., some use hyperspectral, some use elevation, etc.) and each has been implemented with different tools and software, this comparison provides an

Table 5 Comparison between segmentation methods.

Method	OA (%)	Data	Categories
Fuzzy C means[29]	68.9	Aerial image, laser scanning	4 (vegetation, buildings, roads, and open areas)
Segmentation and classification tree method[30]	70	Multispectral aerial imagery	5 (water, pavement, rooftop, bare ground, and vegetation)
Classification trees and test field points[31]	74.3	Aerial image	4 (building, tree, ground, and soil)
Segmentation and classification rules[32]	75	Multispectral aerial imagery	6 (building, hard standing, grass, trees, bare soil, and water)
Region-based GeneSIS[33]	89.86	Hyperspectral image	9 (asphalt, meadows, gravel, trees, metal sheets, bare soil, bitumen, bricks, and shadows)
Object-based imagery analysis[34]	93.17	Aerial orthophotography and DEM	7 (buildings, roads, water, grass, tree, soil, and cropland)
Knowledge-based method[35]	93.9	Multispectral aerial imagery, laser scanning, digital surface models (DSM)	4 (buildings, trees, roads, and grass)
CNN[47]	94.49	Multispectral orthophotography imagery, DSM	5 (vegetation, ground, road, building, and water)
This work	92.63	Satellite images	3 (buildings, vegetation, and roads)

indication of the effectiveness of this method. The OA obtained by the average of the six test cases is used as the comparison metric. The other values are taken from Ref. 47, where the segmentation is performed by training multiple simple neural networks (1 convolution layer and 50 filters) and combining their results. The comparison is shown in Table 5.

It is possible to see that our method outperforms the methods which are not based on deep learning, except for Refs. 34 and 35, which, however, take advantage of a richer dataset with more than one source. As for the results obtained with CNNs in Ref. 47, the difference is certainly due to the fact that our neural network was trained with much less data. This is an essential aspect in deep learning, and in future studies, we plan to increase the size of our dataset. The results, however, are very promising even with the limitations that have been presented.

A further comparison is also made with work presented in Ref. 17, where the same dataset that we presented is employed. The authors tested eight different machine learning methods (fine decision tree, medium decision tree, fine KNN, coarse KNN, cubic KNN, bagged tree, boosted tree, and RUS boosted tree) to segment the satellite images. The total accuracy that the authors achieved is 93.7, which is comparable with the results obtained here. As far as the single images are concerned, our method outperforms Ref. 17 only in some cases, in particular, for images 2, 4, 5, and 6. However, the results suggest that our method, with the right parameter tuning, can outperform state-of-the-art methods.

5 Conclusion and Future Works

Deep learning is receiving growing interest from the academic community, and the availability of more powerful hardware allows for the development of complex applications. Among these, semantic segmentation is undoubtedly one of the most popular and challenging. Unfortunately, accessing the required amount of data combined with good quality labeled ground truth for a high-accuracy neural network is not feasible for everyone. In this work, we applied semantic segmentation to different satellite images representing urban scenes with different proportions of buildings, vegetation, and roads, using a small dataset compared to the ones used in the same field. A CNN based on SegNet was employed using this dataset which we expanded with "hard" augmentation.

The results show promising performance of the network. The scarcity of the dataset does not prevent the network from having high test accuracy, especially for some images, as it did not tend to produce overfitting during the training phase. Moreover, our model is very lightweight, resulting in fast inference with respect to more complex neural networks. The authors believe that even better performances can be achieved with more data.

A second contribution of this work was to show how, in the presence of a small dataset, the variation of the number of layers and filters affect the performance. This knowledge is useful when a small amount of data is available.

References

1. M. Mohammadi et al., "Deep learning for IoT big data and streaming analytics: a survey," *IEEE Commun. Surv. Tutor.* **20**(4), 2923–2960 (2018).
2. N. Neverova, "Deep learning for human motion analysis," Doctoral Dissertation, Université de Lyon (2016).
3. M. Bai and R. Urtasun, "Deep watershed transform for instance segmentation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 5221–5229 (2017).
4. J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 3431–3440 (2015).
5. H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vision*, pp. 1520–1528 (2015).
6. V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: a deep convolutional encoder–decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(12), 2481–2495 (2017).

7. A. G. Howard et al., "Mobilenets: efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861 (2017).
8. S. Hijazi, R. Kumar, and C. Rowen, *Using Convolutional Neural Networks for Image Recognition*, Cadence Design Systems Inc., San Jose, California (2015).
9. J. Gu et al., "Recent advances in convolutional neural networks," *Pattern Recognit.* **77**, 354–377 (2018).
10. K. O'shea and R. Nash, "An introduction to convolutional neural networks," arXiv:1511.08458 (2015).
11. C. Szegedy et al., "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 1–9 (2015).
12. G. Ros et al., "The SYNTHIA dataset: a large collection of synthetic images for semantic segmentation of urban scenes," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, pp. 3234–3243 (2016).
13. C. Couprie et al., "Indoor semantic segmentation using depth information," in *Int. Conf. Learn. Represent.* (2013).
14. A. Garcia-Garcia et al., "A review on deep learning techniques applied to semantic segmentation," arXiv:1704.06857 (2017).
15. G. Ros et al., "Vision-based offline–online perception paradigm for autonomous driving," in *IEEE Winter Conf. Appl. Comput. Vision*, IEEE, pp. 231–238 (2015).
16. S. Muruganandham, "Semantic segmentation of satellite images using deep learning," 2016, https://core.ac.uk/display/81012184.
17. M. C. Younis, E. Keedwell, and D. Savic, "An investigation of pixel-based and object-based image classification in remote sensing," in *Int. Conf. Adv. Sci. Eng. (ICOASE)*, IEEE, pp. 449–454 (2018).
18. D. Masko and P. Hensman, *The Impact of Imbalanced Training Data for Convolutional Neural Networks*, School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm (2015).
19. M. Wieland et al., "Object-based urban structure type pattern recognition from Landsat TM with a support vector machine," *Int. J. Remote Sens.* **37**(17), 4059–4083 (2016).
20. C. Ding et al., "Automatic kernel size determination for deep neural networks based hyperspectral image classification," *Remote Sens.* **10**(3), 415 (2018).
21. D. Soekhoe, P. Van Der Putten, and A. Plaat, "On the impact of data set size in transfer learning using deep neural networks," in *Int. Symp. Intell. Data Anal.*, Springer, Cham, pp. 50–60 (2016).
22. T. Athanasiadis et al., "Semantic image segmentation and object labeling," *IEEE Trans. Circuits Syst. Video Technol.* **17**(3), 298–312 (2007).
23. B. Mičušík and J. Košecká, "Semantic segmentation of street scenes by superpixel co-occurrence and 3D geometry," in *IEEE 12th Int. Conf. Comput. Vision Workshops, ICCV Workshops*, IEEE, pp. 625–632 (2009).
24. J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *IEEE Conf. Comput. Vision and Pattern Recognit.*, IEEE, pp. 1–8 (2008).
25. J. Shotton et al., "Real-time human pose recognition in parts from single depth images," in *CVPR*, IEEE, Vol. 2, pp. 1297–1304 (2011).
26. A. K. Bhandari et al., "Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy," *Expert Syst. Appl.* **41**(7), 3538–3560 (2014).
27. A. K. Bhandari, A. Kumar, and G. K. Singh, "Modified artificial bee colony based computationally efficient multilevel thresholding for satellite image segmentation using Kapur's, Otsu and Tsallis functions," *Expert Syst. Appl.* **42**(3), 1573–1601 (2015).
28. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, Cambridge, Massachusetts (2016).
29. P. Gamba and B. Houshmand, "Joint analysis of SAR, LIDAR and aerial imagery for simultaneous extraction of land cover, DTM and 3D shape of buildings," *Int. J. Remote Sens.* **23**(20), 4439–4450 (2002).

30. N. Thomas, C. Hendrix, and R. G. Congalton, "A comparison of urban mapping methods using high-resolution digital imagery," *Photogramm. Eng. Remote Sens.* **69**(9), 963–972 (2003).
31. L. Matikainen and K. Karila, "Segment-based land cover mapping of a suburban area—comparison of high-resolution remotely sensed datasets using classification trees and test field points," *Remote Sens.* **3**(8), 1777–1804 (2011).
32. C. Sanchez Hernandez, C. Gladstone, and D. Holland, "Classification of urban features from Intergraph's Z/I Imaging DMC high resolution images for integration into a change detection flowline within Ordnance Survey," in *Proc. 2007 IEEE Urban Remote Sens. Joint Event, URBAN 2007-URS* (2007).
33. S. Mylonas et al., "A region-based genesis segmentation algorithm for the classification of remotely sensed images," *Remote Sens.* **7**(3), 2474–2508 (2015).
34. X. Li and G. Shao, "Object-based land-cover mapping with high resolution aerial photography at a county scale in midwestern USA," *Remote Sens.* **6**(11), 11372–11390 (2014).
35. M. J. Huang et al., "A knowledge-based approach to urban feature classification using aerial imagery with lidar data," *Photogramm. Eng. Remote Sens.* **74**(12), 1473–1485 (2008).
36. N. Romero Aquino et al., "The effect of data augmentation on the performance of convolutional neural networks," in *Braz. Soc. Comput. Intell.*, Niterói, Rio de Janeiro (2017).
37. P. V. Tran, "A fully convolutional neural network for cardiac segmentation in short-axis MRI," arXiv:1604.00494 (2016).
38. C. Kamphuis, "Automatic segmentation of retinal layers in optical coherence tomography using deep learning techniques," Master's Thesis, Computing Science—Data Science, Radboud University (2018).
39. "Semantic segmentation using deep learning- MATLAB & Simulink," *Mathworks*, 2018, https://www.mathworks.com/help/vision/examples/semantic-segmentation-using-deep-learning.html.
40. S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," arXiv:1502.03167 (2015)
41. H. U. Jang et al., "Fingerprint spoof detection using contrast enhancement and convolutional neural networks," in *Int. Conf. Inf. Sci. Appl.*, Springer, Singapore, Vol. 424, No. 1, pp. 331–338 (2017)
42. M. I. Sameen, B. Pradhan, and O. S. Aziz, "Classification of very high resolution aerial photos using spectral–spatial convolutional neural networks," *J. Sens.* **2018**, 1–12 (2018).
43. M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks* **106**, 249–259 (2018).
44. S. A. Taghanaki et al., "Combo loss: handling input and output imbalance in multi-organ segmentation," *Computerized Med. Imaging and Graphics* **75**, 24–33 (2019).
45. A. Atakulreka and D. Sutivong, "Avoiding local minima in feedforward neural networks by simultaneous learning," in *Aust. Joint Conf. Artif. Intell.*, Springer, Berlin, Heidelberg, pp. 100–109 (2007).
46. J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *Biometrics* **33**, 159–174 (1977).
47. M. Längkvist et al., "Classification and segmentation of satellite orthoimagery using convolutional neural networks," *Remote Sens.* **8**(4), 329 (2016).

Biographies of the authors are not available.