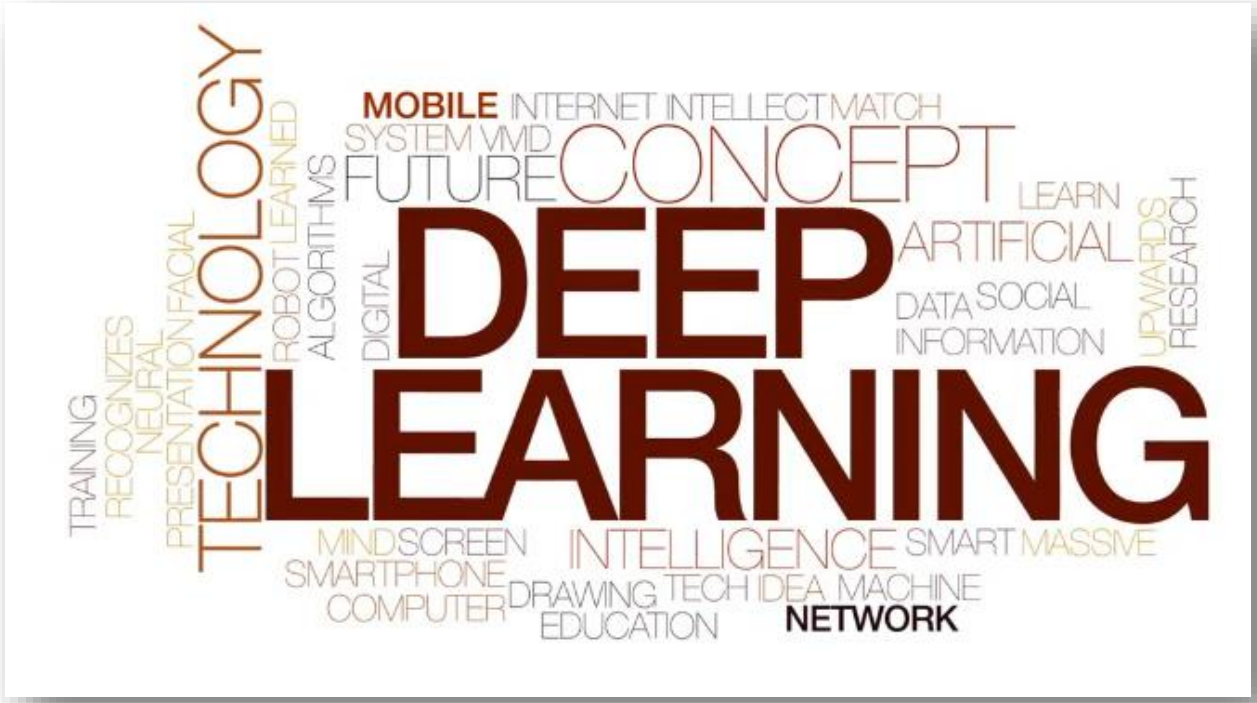


RESEARCH REPORT NO. 2

Use of artificial neural network for image segmentation

Abstract

The segmentation of aerial images is a process relying purely on spectral content, and this makes it a very challenging topic.

In this paper, a convolutional neural network (CNN) we used to perform semantic segmentation of true-orthophoto into six classes, such as roads, buildings, low vegetation, trees, cars, and others. To test the process of segmentation we use two datasets, both ISPRS benchmarks (Vaihingen and Potsdam). We base the architecture of the neural network used on SegNet deep convolutional network and the initial weights used are the ones got using VGG16 on the dataset ImageNet.

In Section 1, called Introduction, we define the notions of the artificial neural network, the natural and artificial neurons, and the network architectures.

In Section 2, we present the methods used, starting with convolutional neural network and ending with SegNet architecture. Section 3 is the study case, where we define the datasets used, and the results got. In Section 4 we conclude our work.

Phd coordinator

Univ. prof. phd. eng. Petre Iuliu Dragomir

Phd student

Phd. Eng. Bină Iuliana Maria (Pârvu)

1. Introduction

We use artificial neural networks a lot nowadays, in almost all fields of technology. Some of the most well-known names in today's technology, such as Google, Facebook, Microsoft, Amazon invest and use artificial intelligence in their own business models. Some well-known applications are virtual assistants (eg Alexa), artificial vision, voice recognition, facial recognition, autonomous translation, etc.

Deep neural networks are used successfully in medicine, for image analysis, drug design, diagnostic system, etc., by the car manufacturers of autonomous vehicles, etc. The basic applications of artificial neural networks, in photogrammetry and remote sensing, refer to object detection (Fig. 1), image segmentation or classification (Fig. 2) and change detection over periods in time.

Fig. 1. Object detection using the software Visual Profiler [1]

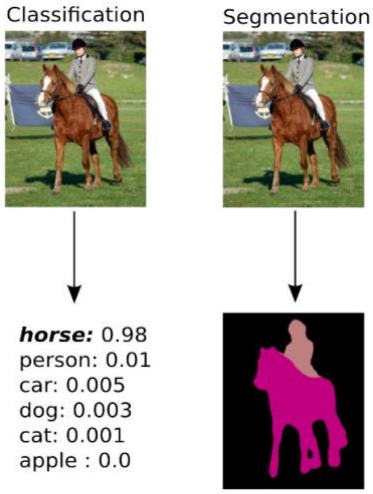


Fig. 2. Image segmentation or classification [2]

Haykin defines a neural network as "a massively parallel distributed processor made up of simple processing units that has a natural propensity for storing experiential knowledge and making it available for use" [3].

1.1.Basic Principles

The brain neuron is an electrically excitable cell that communicates with other cells through specialized connections called synapses, being the morpho-functional unit of the nervous system. There are 100 billion neurons in a human brain, arranged in a network. Each neuron has about 7000 synapses with other neurons, so the number of synapses for an adult varies between 10^{14} and $5x10^{14}$.

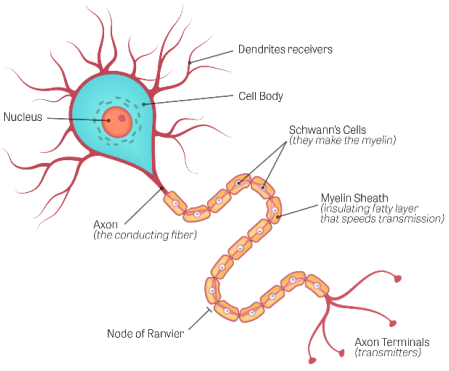


Fig. 3. Human neuron structure (Source: iStock Photos, ©)

The basic components of an artificial neuron are:

- ✓ a set of connections/synapses between units (each connection is characterized by a weight);
- ✓ a summing function that gathers the input signals, weighted with the synaptic power of the respective neuron;
- ✓ an activation function, with the role of limiting the allowable amplitude range of the output signal to a certain finite value;
- ✓ a bias applied from outside the network which aims to increase or decrease the net input of the activation function, depending on its value (positive or negative).

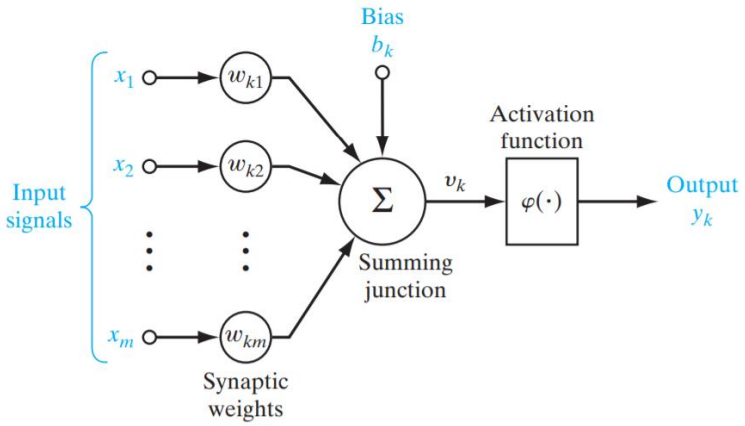


Fig. 4. Nonlinear model of an artificial neuron (labeled *k*) [3]

1.2. Artificial neural network architectures

The way the neurons of an artificial neural network are structured is linked with the learning algorithm used to train the network. In the following, we present three different network architectures or structures.

a. Single-layer feedforward network

A feedforward network causes the input layer to be projected directly on the output layer of neurons, but not vice versa. Single-layer refers to the fact that the network performs the computation in a single layer.

b. Multi-layer feedforward network

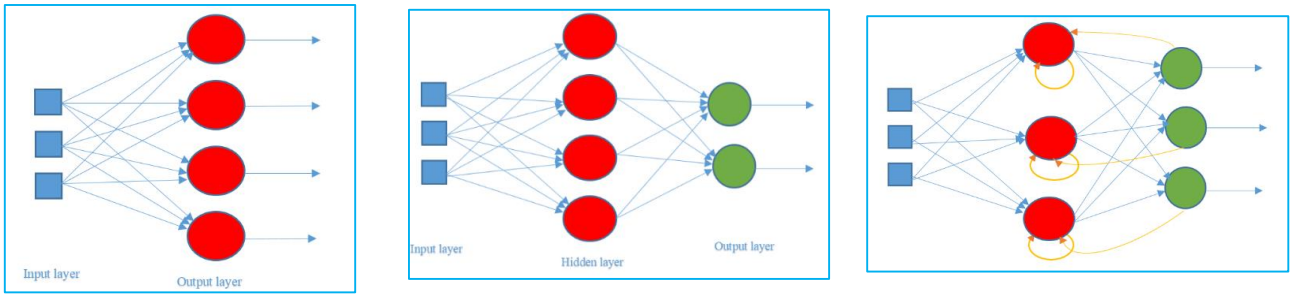
A multi-layer feedforward network has one or more hidden layers, in which the computation nodes are called hidden units. These layers achieve improved results. We can say that the network gains a global perspective, because of the additional set of synaptic connections and the additional size of neural interactions [4].

Multi-layer feedforward networks are slower but can implement more complex functions than single-layer networks.

c. Recurrent Networks

A recurrent/feedback neural network has at least one feedback loop. The simplest recurrent architecture comprises a layer of neurons, where each neuron transmits its output signal back to the inputs of all other neurons.

Feedback loops involve the use of time or delay elements on these branches, which causes nonlinear-dynamic behavior. This architecture is the most complex and depending on the density of reverse connections, there are total recurrent networks (Hopfield model) and partial recurrent networks (Elman or Chua-Yang model) [5].



a. single-layer feedforward network b. multi-layer feedforward network c. recurrent network

Fig. 5. Artificial neural network architectures

2. Methods

2.1.Convolutional neural network (CNN)

CNN is a deep neural network, composed of a convolutional part, which extracts the specific characteristics of the input data and stores them as a vector and fully connected layers - type Perceptron multilayer, which perform the classification part [6].

The first application of convolutional neural networks is the LeNet-5 architecture, introduced by Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner, in 1998. They developed it for hand letters recognition in documents (Fig. 6.) and it is still used, successfully, in the image classification process.

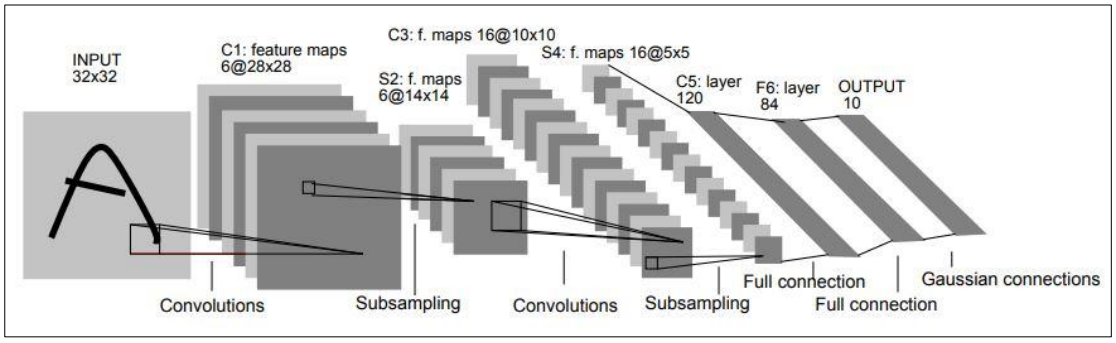


Fig. 6. LeNet-5 architecture [7]

To better describe the structure of a CNN we have to explain some notions: convolutional layer, pooling layer, and fully connected layers.

Convolutional layers are based on the mathematical operation by which two functions produce a third one which expresses how the shape of the first changes the second. This layer reduces the size of the input image for further processing. It scrolls the image from left to right and from bottom to top with a kernel filter and it computes the corresponding values in the convolution layer by adding the multiplied values in the filter by the original values of the pixels in the image. After scrolling the entire image, we get the activation or feature map.

The role of the *pooling layer* is to further reduce the spatial size of the image, the processing time, and to highlight certain features. The hyper-parameters used are the size of the pooling filter and the step of sliding it over the input matrix. The two types of pooling applied on CNNs are max pooling and average pooling.

The *fully connected layer* has to adapt the output of the convolutional part to a vector with N dimensions, where N is the number of classes used in the classification process. The values in this vector represent the probability value of a certain class.

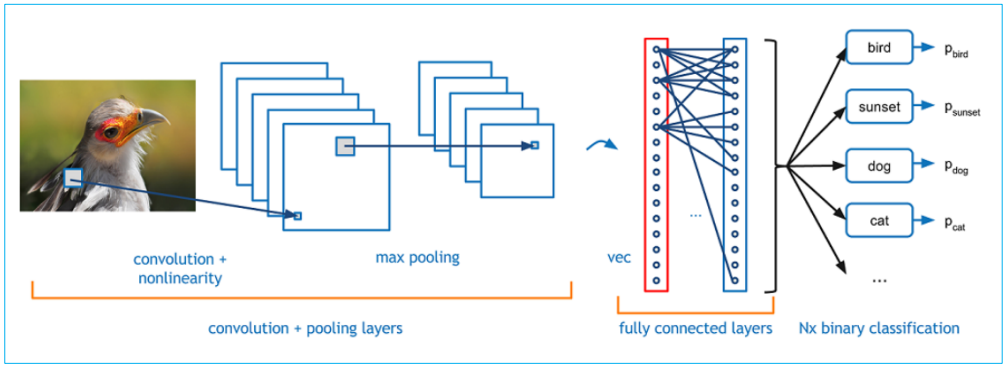


Fig. 7. CNN arhitecture [8]

2.2.SegNet

SegNet is a deep convolutional neural network, developed within the Computer Vision and Robotics Group at Cambridge University [9] and used for pixel-level semantic segmentation.

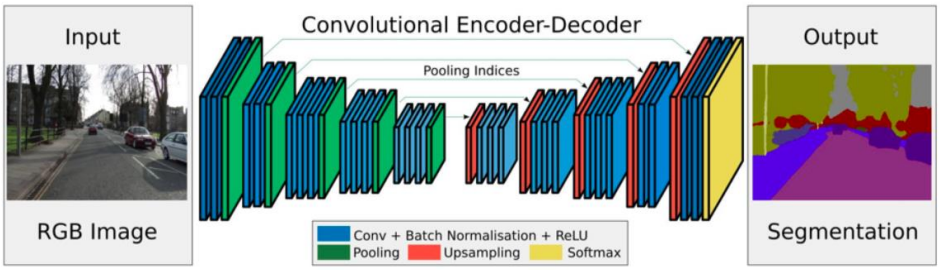


Fig. 8. SegNet arhitecture [9]

The network comprises layers of encoders, decoders, and a final layer of pixel-level classification.

The network of encoders represented in Fig. 9, comprises 13 convolutional layers (corresponding to VGG16 network). Each encoder layer has a corresponding decoder layer. If in the encoding phase, we do a sub-sampling of the data, in the decoding part we apply the reverse operation - over-sampling, so that the final image will have the same size as the initial one.

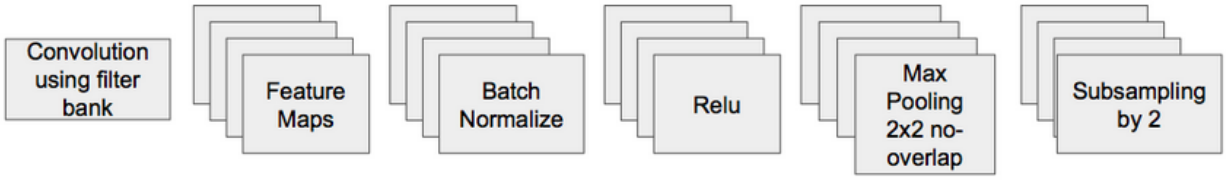


Fig. 9. Network of encoders [10]

To test SegNet architecture, we used the demonstration product, from the website *http://mi.eng.cam.ac.uk/projects/segnet/demo.php#demo* on a randomly selected image from Romania. We can see the result got for the 12 classes used for segmentation in Fig. 10.

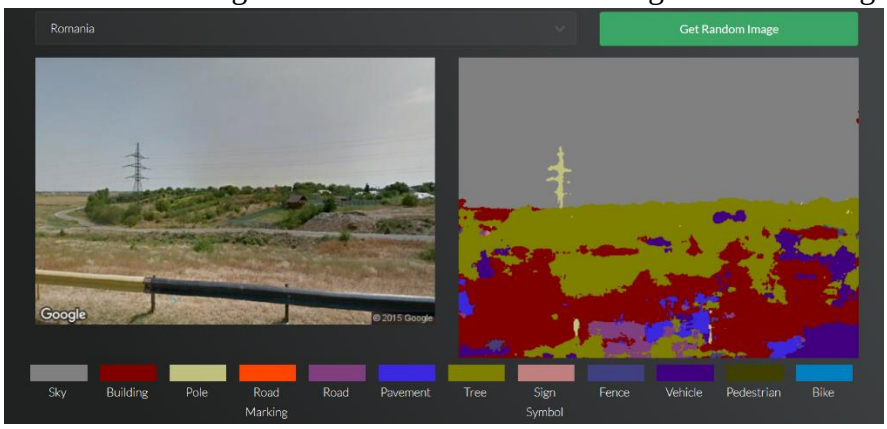


Fig. 10. Image semantic segmented using SegNet [11]

3. Study case

3.1. Datasets used in the study case

The first images used come from the subset of data acquired over the city of Vaihingen an der Enz, Germany, by the German Association for Photogrammetry and Remote Sensing [12]. They purchased the images with the RWE Power Intergraph/ZI photogrammetric camera in the summer of 2008 and have the following features:

- ground sample distance - 8 cm;
- radiometric resolution - 11 bites;
- spectral resolution - IR, R, G;
- number of images - 38.

The generated orthophotos are TOP type (true-orthophoto), with a spatial resolution of 9 cm. The dataset used comprises true-orthophotos, ground-truth images, and eroded ground-truth images.

The classes used in the segmentation process are ROADS, BUILDINGS, LOW VEGETATION, TREES, CARS, and OTHERS.

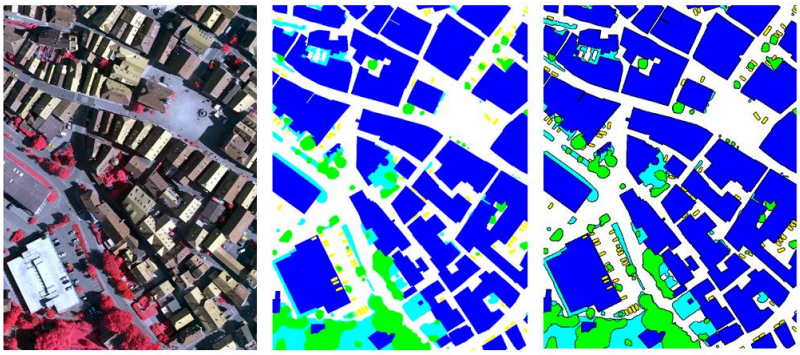


Fig. 11. Example from the Vaihingen dataset

The second images used come from the subset of data acquired over the city of Potsdam, Germany, by BSF swissphoto [13] and have the following features:

- ground sample distance - 5 cm;
- radiometric resolution - 8 bites;
- spectral resolution - IR, R, G / R, G, B / R,G, B, IR;
- number of images - 38.

The generated orthophotos are TOP type, with a spatial resolution of 5 cm. The classes used in the segmentation process are the same as in the Vaihingen dataset.

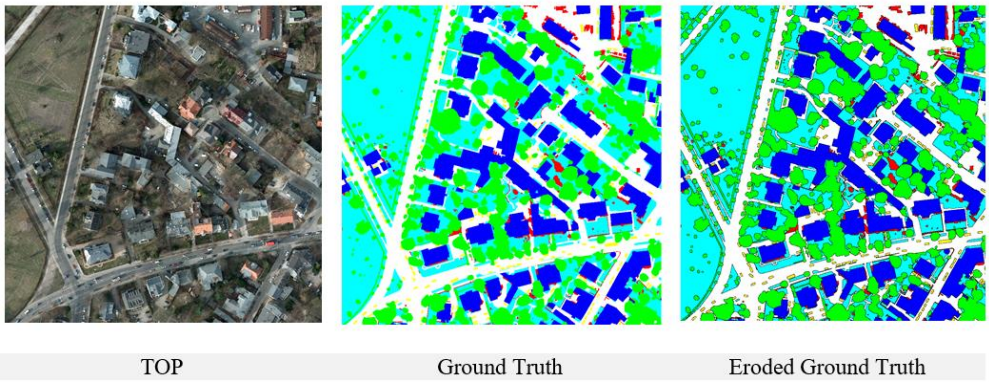


Fig. 12. Example from the Potsdam dataset

3.2.The arhitecture of the neural network used for image segmentation

To test image segmentation using artificial neural networks, we used the Python code available at *https://blesaux.github.io/teaching/DL4RS* [14]. We run the code in the Google Drive environment. It is an implementation of the Pytorch library for segmenting aerial images using the SegNet architecture [15].

Initially, in the code we set the parameters (different depending on each dataset), such as window size, step, classes used in segmentation, the dataset used. If we store the Ground Truth image in RGB format, we need to define the color palette that can map the class ID to the color in

the RGB palette. We must define the auxiliary functions in the code (eg: definition of the confusion matrix, F1 score, kappa coefficient). Then, we load all images into memory and random sample them. After this, we define the SegNet architecture. We initialize the network and weights using the He approach. Optionally, we can load pre-trained weights; we used the weights determined for the ImageNet dataset using VGG16 architecture. We loaded the network on the GPU to facilitate the training and testing process and we divide the data into learning and test set. In the optimization process, we use the Stochastic Gradient Descent algorithm to optimize the weights, and we train the encoder at half of the decoder learning rate. The next step is to train the network, and we do this in one or more epochs. After we complete the training process, we load the final weights and we test the network using a reasonable step. The results comprise classified images and accuracy indicators.

3.3.Results

To test and compare the results, we used different subsets of the data: two tries for Vaihingen and one for Potsdam. For these three datasets, we ran the code using a different number of epochs.

Test 1 - Vaihingen (2 images for training and one image for testing)

The images used for training are top_mosaic_09cm_area1.tif and top_mosaic_09cm_area23.tif and the image used for testing the network is top_mosaic_09cm_area5.tif.

We present the confusion matrix and the accuracy indicators obtained after the training process in Table 1 and Table 2. It is important to specify that we run only one epoch.

Tabel 1. Confusion matrix after the training process

CLASSES	ROADS	BUILDINGS	LOW VEGETATION	TREES	CARS
ROADS	1128357	323097	56151	7899	293
BUILDINGS	192253	2085525	19065	3860	1847
LOW VEGETATION	17399	3690	112131	130589	0
TREES	5691	685	9178	277082	0
CARS	32731	8498	2057	529	1131

Tabel 2. Accuracy indicators after the training process

CLASSES	F1 SCORE (%)	OA (%)	KAPPA (%)
ROADS	78.02	81.55	69.18
BUILDINGS	88.29		
LOW VEGETATION	48.50		
TREES	77.76		
CARS	4.69		
OTHERS	-		

After we finish the training process, we load the final weights, and we test the network using a reasonable step. We detail the values of the accuracy indicators resulted after the testing process in Table 3.

Tabel 3. Accuracy indicators after the testing process

CLASSES	F1 SCORE (%)	OA (%)	KAPPA (%)
ROADS	78.90	82.14	70.14
BUILDINGS	**88.62**		
LOW VEGETATION	49.39		
TREES	78.13		
CARS	3.83		
OTHERS	-		

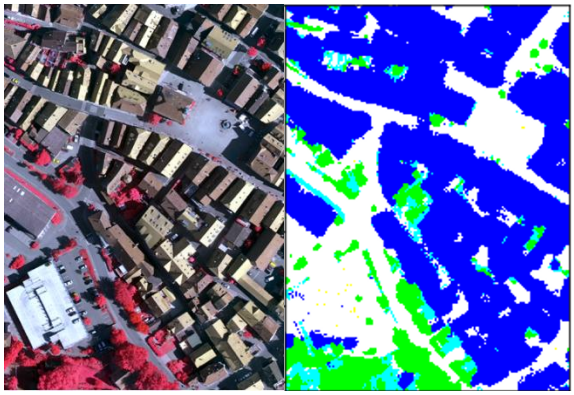


Fig. 13. TOP *top_mosaic_09cm_area5.tif* and the corresponding result (one epoch)

We ran the entire process for 2 and 10 epochs, and we display the results got in Table 4. We can observe an improvement in the accuracy indicators that is directly proportional to the number of epochs used in the training process. We detail the F1 score for the BUILDINGS class in Table 4. We segmented the buildings optimally for 10 epochs, with an F1 score equal to 91.76%.

Tabel 4. Accuracy indicators for Test 1

Number of epochs	OA (%)	F1 SCORE (%) - BUILDINGS	KAPPA (%)	TRAINING PROCESS TIME (MIN)
1	82.14	88.61	70.13	10
2	85.98	91.06	76.55	15
10	87.30	**91.76**	79.19	60

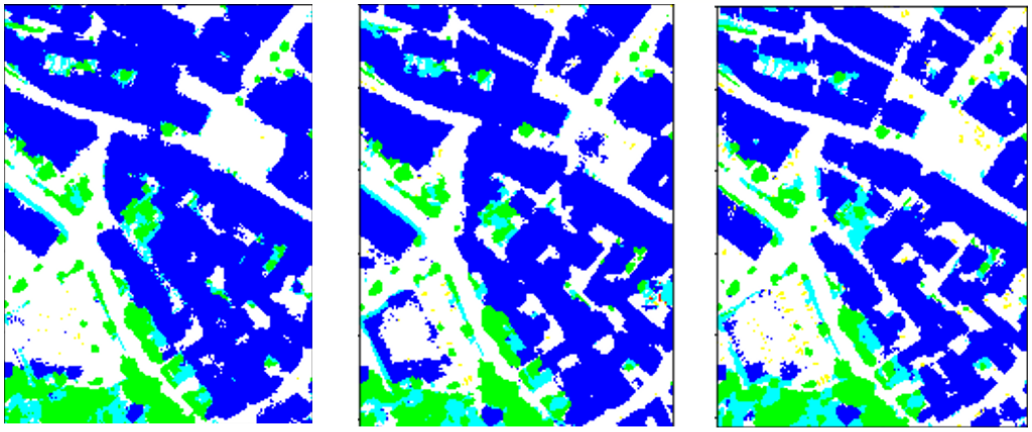


Fig. 14. Segmentation results for Test 1

Test 2 - Vaihingen (4 images for training and two images for testing)

The images used for training are top_mosaic_09cm_area5.tif, top_mosaic_09cm_area21.tif, top_mosaic_09cm_area23.tif and top_mosaic_09cm_area28.tif and the images used for testing the network are top_mosaic_09cm_area7.tif and top_mosaic_09cm_area30.tif.

Tabel 5. Accuracy indicators for Test 2

CLASSES	2 EPOCHS			10 EPOCHS			20 EPOCHS		
	F1 SCORE (%)	OA (%)	KAPPA (%)	F1 SCORE (%)	OA (%)	KAPPA (%)	F1 SCORE (%)	OA (%)	KAPPA (%)
ROADS	82.21	78.09	70.74	87.83	86.20	81.63	87.53	85.76	81.03
BUILDINGS	**84.32**			**91.83**			**91.27**		
LOW VEGETATION	58.43			75.37			74.64		
TREES	81.12			87.10			86.71		
CARS	4.93			72.96			71.75		
OTHERS	-			-			-		

The values of the accuracy indicators follow an upward trend until the use of approximately 12 epochs, then the trend becomes downward (Fig. 15). We can conclude that for the subset used, we got the optimal results by training the network with 10 epochs.

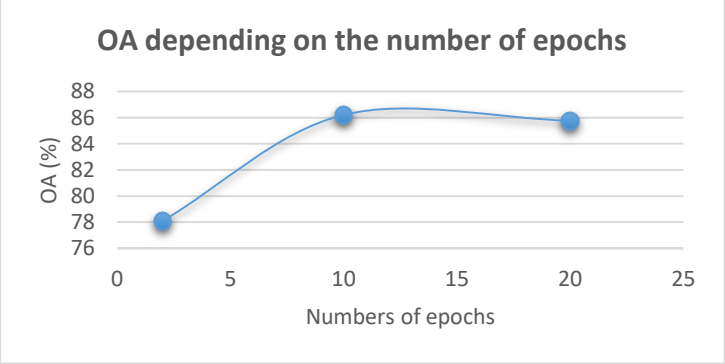


Fig. 15. Graph of the overall accuracy

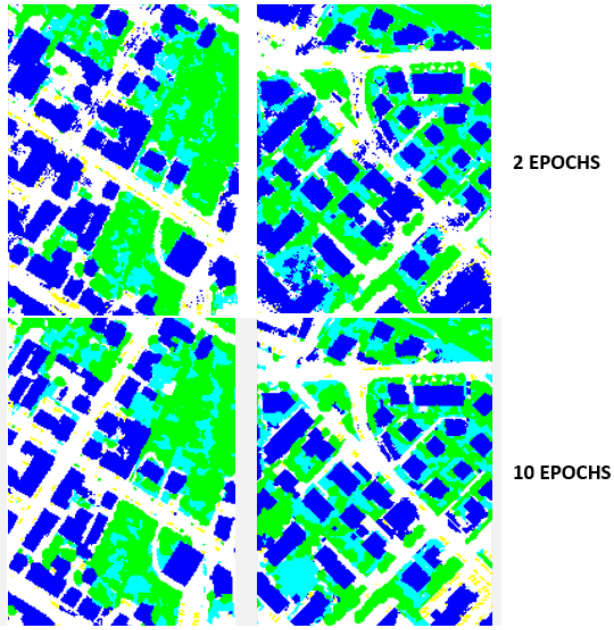


Fig. 16. Segmentation results for Test 2

Test 3 - Potsdam (3 images for training and one image for testing)

The images used for training are top_potsdam_2_10_RGB.tif, top_potsdam_2_11_RGB.tif and top_potsdam_2_13_RGB.tif and the image used for testing the network is top_potsdam_2_14_RGB.tif.

We display the confusion matrix and the values of the accuracy indicators in the training process, using 5 epochs, in Table 6.

Tabel 6. Confusion matrix after the training process

CLASSES	ROADS	BUILDINGS	LOW VEGETATION	TREES	CARS	OTHER
ROADS	**3705817**	663047	501458	141697	2591	10234
BUILDINGS	90835	**1711595**	169894	41510	8754	331776
LOW VEGETATION	1234399	46820	**11660323**	1074105	1559	5260
TREES	737909	245355	3751823	**7416668**	12769	27179
CARS	13377	44144	7029	1512	**85334**	9178
OTHER	7661	35768	22244	4284	11085	**5322**

After analyzing the values in the confusion matrix, we observe the following problems:

- ✓ the OTHER class generates the biggest confusions in the learning process;
- ✓ there are some significant confusions between the ROADS and LOW VEGETATION classes;
- ✓ the CARS class is not well differentiated from the other classes.

Network training is a long process; in this subset and depending on the number of epochs, the time varied between one hour and 24 hours.

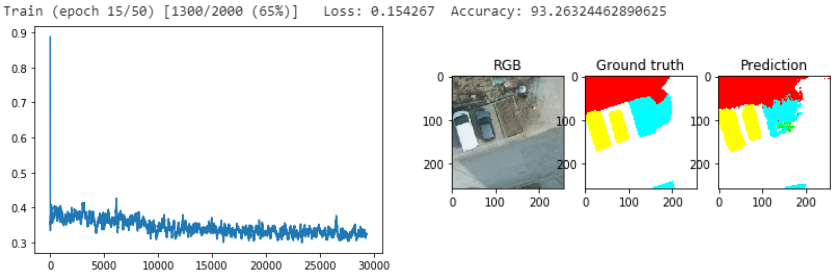


Fig. 17. Snapshot from the training process

In Fig. 17 we can observe an on-the-fly accuracy of 93.26% for the patch in the image, within the epoch number 15, out of 50 epochs.

As part of the optimization algorithm, we have to repeatedly estimate the error for the current state of the model. This requires an error function, conventionally called the loss function, which can estimate the loss in the model so that we can update the weights to reduce the loss at the next evaluation. The function we want to minimize or maximize is called objective function or criterion. If we want to minimize this function, we get the cost or loss function [16].

Tabel 7. F1 Score for Test 3

CLASSES	F1 SCORE (%)				
	5 EPOCHS	10 EPOCHS	20 EPOCHS	30 EPOCHS	50 EPOCHS
ROADS	71.14	82.26	89.08	88.68	86.33
BUILDINGS	**70.75**	**77.06**	**94.22**	**95.06**	**95.52**
LOW VEGETATION	77.95	81.66	88.42	88.07	86.41
TREES	71.89	79.81	87.66	85.91	85.16
CARS	61.04	71.30	87.90	86.66	89.09
OTHER	2.30	0.58	14.79	18.65	22.14

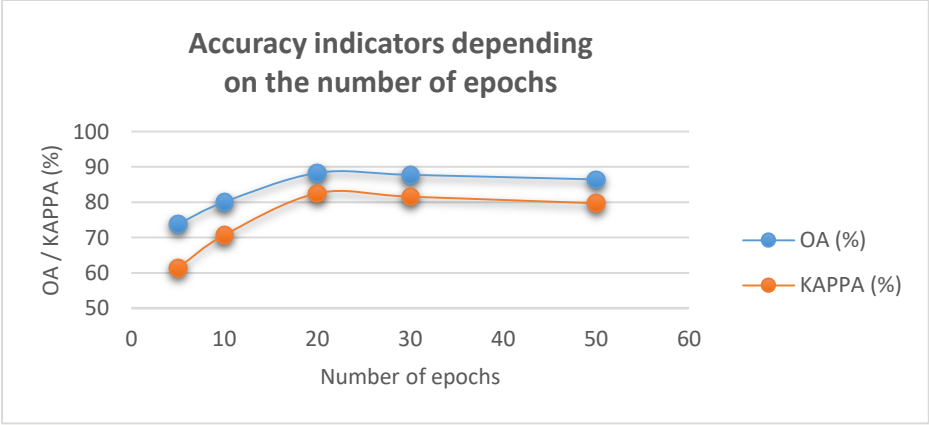
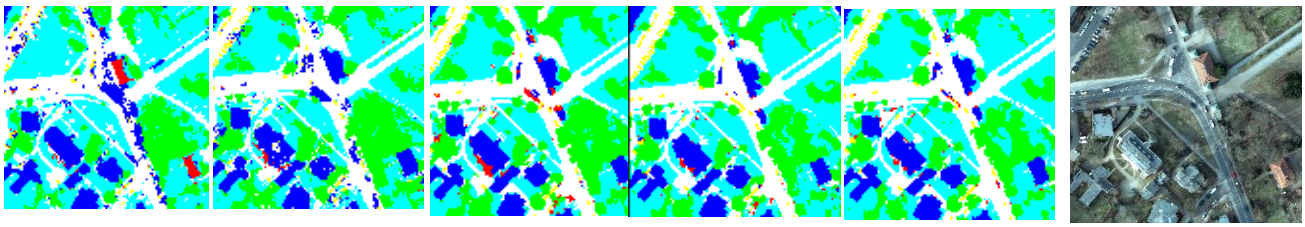


Fig. 18. Graph of the accuracy indicators for Test 3

Based on the information Fig. 17 and Fig. 18, we can observe a decrease in the segmentation accuracy after 20 epochs, but an improvement in the F1 score for the BUILDINGS

class. For buildings we can state that we got the best results after training the network on 50 epochs. The time required to run the training process for 50 epochs was 6 hours.



5 epochs | 10 epochs | 20 epochs | 30 epochs | 50 epochs | RGB image

Fig. 19. Segmentation results (detail image)

4. Conclusions

Nowadays, AlexNet architecture [17] has been overtaken by much more efficient architectures, but it is the key step towards today's deep neural networks.

We mainly use deep neural networks today, because we have met the three basic criteria: the data sets comprise big data, the hardware capabilities given by the GPU are impressive and the software component presents improved techniques and new models.

We base the choice of the architectural model used and the training of the network on attempts and personal experience. Of all the stages of a complex process using neural networks, the training process has the longest duration.

One of the basic advantages of neural networks is the ability to model both linear and nonlinear functions.

Neural networks can perform well even if the system is damaged, because of the large number of neurons.

Convolutional neural networks have become a standard in applications based on image analysis, their performance being able to equal human performance in some fields of activity. To get good results we recommend it to use small size filters (3 × 3 or 5 × 5), pitch equal to 1 and padding with zero so it does not alter the spatial dimension of the input image.

The values in the resulting confusion matrix provide valuable information about the existing ambiguities in class differentiation.

For Test 1 - Vaihingen, we segmented the buildings best after 10 epochs of network training, with an F1 score of 91.76%.

For Test 2 - Vaihingen, we segmented the objects best after 10 epochs of network training, for the BUILDINGS class the F1 score was 91.83%, for the ROADS class the F1 score was 87.83%, and for the TREES class the F1 score was 87.1%. From the graph shown in Fig. 15, we conclude that we can get the best results for 12 epochs; after increasing this number of epochs, the accuracy of the segmentation decreases.

For Test 3 - Potsdam, the OTHER class generates the biggest confusion in the learning process, for this reason the F1 score got for this class varies between 2.3% and 22.14%, depending

on the number of epochs. For this dataset there is a decrease in the segmentation accuracy after 20 epochs, but an improvement in the F1 score for the BUILDINGS class. Thus, for buildings we got the best results after 50 epochs (F1 score - 95.52%).

Acknowledgments

The German Society for Photogrammetry, Remote Sensing and Geoinformation (DGPF) provided the Vaihingen dataset. The BSF Swissphoto provided the Potsdam dataset. To test image segmentation using artificial neural networks, we used Python code available at *https://blesaux.github.io/teaching/DL4RS*.

References

1. https://www.video-inform.com/visual-profiler/, Visited: 15.02.2020
2. Lefèvre, S., Le Saux, B., Landrieu, L.: DEEP LEARNING FOR REMOTE SENSING - EDUSERV COURSE, EuroSDR, 2019
3. Haykin, S.: NEURAL NETWORKS AND LEARNING MACHINES, Third Edition, Pearson, ISBN-13: 978-0-13-147139-9, ISBN-10: 0-13-147139-2, 2008
4. Churchland, P. S. & Sejnowski, T. J.: THE COMPUTATIONAL BRAIN. Computational neuroscience. The MIT Press, 1992
5. http://staff.fmi.uvt.ro/~daniela.zaharie/cne2014/curs/cne2014_slides5.pdf, Visited 14.01.2020
6. Vrejoiu, M. H.: REŢELE NEURONALE CONVOLUŢIONALE, BIG DATA ŞI DEEP LEARNING ÎN ANALIZA AUTOMATĂ DE IMAGINI, Romanian Journal of Information Technology and Automatic Control, Vol. 29, No. 1, pp. 91-114, 2019
7. LeCun, Y., Bottun, L., Bengio, Y., Haffner, P.: GRADIENT-BASED LEARNING APPLIED TO DOCUMENT RECOGNITION, *Proceedings of the IEEE, V*ol. 86, No. 11, pp. 2278-2324, 1998
8. https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/, Visited 20.01.2020
9. Badrinarayanan, V., Kendall, A., Cipolla, R.: SEGNET: A DEEP CONVOLUTIONAL ENCODER-DECODER ARCHITECTURE FOR IMAGE SEGMENTATION, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 12, pp. 2481-2495, doi: 10.1109/TPAMI.2016.2644615, 2017
10. https://towardsdatascience.com/summary-of-segnet-a-deep-convolutional-encoder-decoder-architecture-for-image-segmentation-75b2805d86f5, Visited 09.02.2020
11. http://mi.eng.cam.ac.uk/projects/segnet/demo.php#demo, Visited 09.02.2020

12. Cramer, M.: THE DGPF TEST ON DIGITAL AERIAL CAMERA EVALUATION – OVERVIEW AND TEST DESIGN, Photogrammetrie – Fernerkundung – Geoinformation 2 (2010), pp. 73-82, 2010

13. http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-potsdam.html, Visited: 16.02.2020

14. https://blesaux.github.io/teaching/DL4RS, Visited 25.02.2020

15. Audebert, N., Le Saux, B., Lefèvre, S.: BEYOND RGB: VERY HIGH RESOLUTION URBAN REMOTE SENSING WITH MULTIMODAL DEEP NETWORKS, ISPRS Journal of Photogrammetry and Remote Sensing, Elsevier, Vol. 140, pp.20-32, 2018

16. Goodfellow, I., Bengio, Y., Courville, A.: DEEP LEARNING (ADAPTIVE COMPUTATION AND MACHINE LEARNING SERIES), MIT Press, 2017

17. Krizhevsky, A., Sutskever, I. & Hinton, G. E.: IMAGENET CLASSIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS, Advances in neural information processing systems, Vol. 25(2), DOI: 10.1145/3065386, 2012