Patch-based Deep Learning Architectures for Sparse Annotated Very High Resolution Datasets

Maria Papadomanolaki[a], Maria Vakalopoulou[a b], Konstantinos Karantzalos[a]
[a] Remote Sensing Laboratory, National Technical University of Athens, Greece
[b] Center for Visual Computing, CentraleSupélec, Inria, Université Paris-Saclay, France
E-mail: mar.papadomanolaki@gmail.com, mariavak@central.ntua.gr, karank@central.ntua.gr

***Abstract*—In this paper, we compare the performance of different deep-learning architectures under a patch-based framework for the semantic labeling of sparse annotated urban scenes from very high resolution images. In particular, the simple convolutional network *ConvNet*, the *AlexNet* and the *VGG* models have been trained and tested on the publicly available, multispectral, very high resolution Summer Zurich v1.0 dataset. Experiments with patches of different dimensions have been performed and compared, indicating the optimal size for the semantic segmentation of very high resolution satellite data. The overall validation and assessment indicated the robustness of the high level features that are computed with the employed deep architectures for the semantic labeling of urban scenes.**

I. Introduction

Urban classification from various sensors is a well studied problem in the remote sensing community [1], [2], [3] with important research and development efforts during the last decades. Depending on the resolution and the type of the remote sensing data several methods have been proposed in the literature, mainly addressing the task with pixel-based or object-based frameworks.

Recently, neural networks with deep architectures have reached state-of-the-art results for image semantic labeling in the computer vision and machine learning communities [4], [5]. They have led to significant classification performances due to their capacity to build powerful high-level features. Autoencoders, Deep Boltzmann Machines, Deep Belief Networks and Convolutional Neural Networks (CNNs) are some of the most commonly used architectures in the literature. Additionally, the remote sensing community has adopted them successfully for addressing several classification tasks with data from various remote sensing sensors [6], [7], [8], [9], [10], [11].

Among them, approaches targeting semantic labeling from VHR images can significantly contribute to efficient semantic segmentation of urban scenes. For example, authors in [11] propose a multiscale patch-based approach for semantic labeling of VHR images which was based on CNN features, hand-crafted features as well as Conditional Random Fields (CRFs). Moreover, a CRF model [12] based on ring-based, class-interaction potentials was recently proposed which improved average class accuracy semantic labeling rates against standard approaches.

In this paper, we compare the performance of three different deep architectures *i.e.,* a simple *ConvNet*, *AlexNet* [13] and *VGG* [14] for semantic labeling on the publicly available multispectral dataset, Summer Zurich v1.0 [12]. The sparse annotated ground truth of this particular dataset was the main reason for benchmarking patch-based and not for example dense prediction approaches. Additionally, we assess the performance of the different patch dimensions *e.g.,* 11x11, 21x21, 29x29, 33x33 and 45x45, by comparing their resulting accuracy rates towards optimal patch size selection for urban semantic labeling. It should be noted that although features are calculated at deeper layers with relatively small patch sizes, which seems to affect their quality and robustness, we did perform experiments in order to evaluate this particular aspect when patches of different sizes are employed.

II. Methodology

For the training process, three different deep learning models have been employed and compared in this paper. Their architecture is based on CNNs and similarly to any neural network, they accept as input a training vector which is then processed by the neurons of internal layers. The most common CNN layers are the convolutional, pooling and activation ones. Convolutional layers play the most significant role in CNNs, as they execute most of the excessive mathematical lifting. They include many learned filters for the extraction of features, which are also known as receptive fields. Many other hyperparameters are available enabling the fine tuning of the models. For example, stride and zero-padding are such parameters which mainly control the number of neurons. In particular, stride represents the step of the filter expressed with number of pixels, while the zero-padding allows the filling of the input volume with zeros at the border. Regarding the activation function layers, they are of various types and each one applies a different function to the processed dataset depending on the training needs. Some of the functions that can be used are tanh, softmax *etc*. Lastly, pooling layers have a downsampling role. Similar to convolutional layers, they are composed of local filters which contribute to the reduction of data, thus leading to a decrease in computational activities.

In the following paragraph, the deep learning models that were employed for the training process are described along with the implementation parameters.

A. *The Simple ConvNet Network (**ConvNet**)*

A relatively simple *ConvNet* network consisting of 4 layers (Figure 1): 2 convolutional and 2 fully connected has been

978-1-5090-5808-2/17/$31.00 © 2017 IEEE

tested. The architecture of this model is displayed in Figure 1 for a patch size of 29x29 pixels. More specifically, the initial patch is given as input to the first convolutional layer, which produces an output volume of size 32x25x25. The tanh function is then applied element-wise to the input tensor. Since *ConvNet* is a relatively narrow model, the tanh function was preferred in this setup due to the symmetrical output that produces (*i.e.,* its range is (-1,1)), which limits biases and avoids saturation. The rectified linear unit (ReLU) function can be used as well. Then a max-pooling operation follows, which downsamples the training dataset and lightens the computational burden. The next convolutional layer follows the same pattern, feeding the output to the last 2 fully-connected layers that produce the distribution over the 8 different classes.

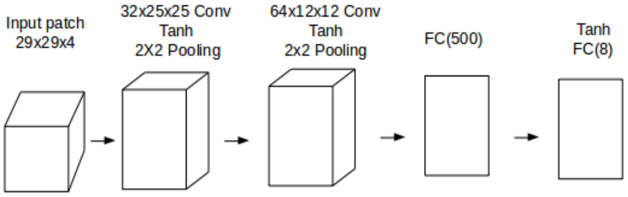


Fig. 1. Graphical illustration of the *ConvNet* model for a 29x29 patch dimension. Dimensions refer to internal layers and represent data sizes, not receptive fields.

B. The AlexNet Network (**AlexNet**)

Moreover, we employed the *AlexNet* architecture [13] for the different patch-based experiments. *AlexNet* is deeper than the *ConvNet*, consisting of 8 layers (Figure 2): 5 convolutional and 3 fully connected. More specifically, the first convolutional layer receives the raw input patch which consists of 4 channels (or input planes) and is of size 29x29 for the specific example. The image is processed by kernels of size 4x3x3 and a stride of 1 pixel, producing an output volume of size 16x27x27. After that, the rectified linear unit (ReLU) function is applied element-wise to the input tensor. Lastly, a max-pooling operation is applied, which reduces the spatial size of the input volume using kernels of size 3 and a stride of 2 producing a final output of size 16x13x13. The next layer follows the same pattern (Convolution-ReLU-MaxPooling) resulting in an output volume of size 96x6x6. The third convolutional layer applies 3x3 kernels and a stride and zero padding of 1, followed by the application of the rectified linear unit function. The fourth layer has the same form (Convolution-ReLU) resulting in an output volume of size 64x6x6. This volume is then processed by the last convolutional layer (Convolution-ReLU-MaxPooling) which feeds the outcome to the last 3 fully connected layers. The final outcome of the model is given to a softmax function which produces the desired results.

C. VGG Network (**VGG**)

In addition, we tested a variation of the relative deeper *VGG* model [14]. One variation between the original *VGG* model and the one implemented in all our experiments is the use of batch-normalisation and dropout layers in the consecutively convolutional layers. In particular, the implemented model consists of 16 layers: 13 convolutional and 3 fully connected. This model repeatedly makes use of convolutions followed by batch-normalisation operations and applications of the rectified linear unit function. If we consider this group of consecutive operations as a function named ConvBNReLU, we can see the form of the entire model in Figure 3. One can observe that there are also many dropout layers, which reduce the possibility of overfitting.

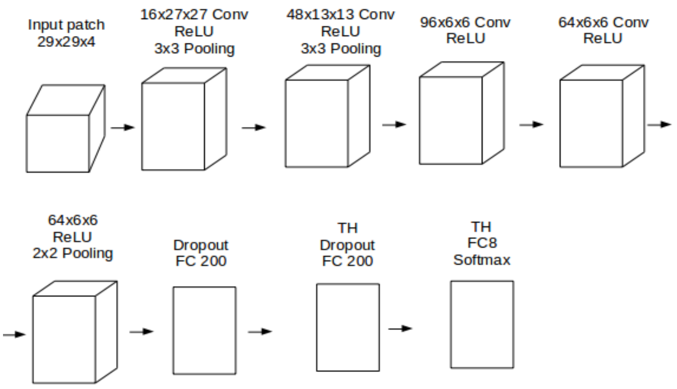


Fig. 2. Graphical illustration of *AlexNet* model for patches of size 29x29. TH represents the application of the threshold function.

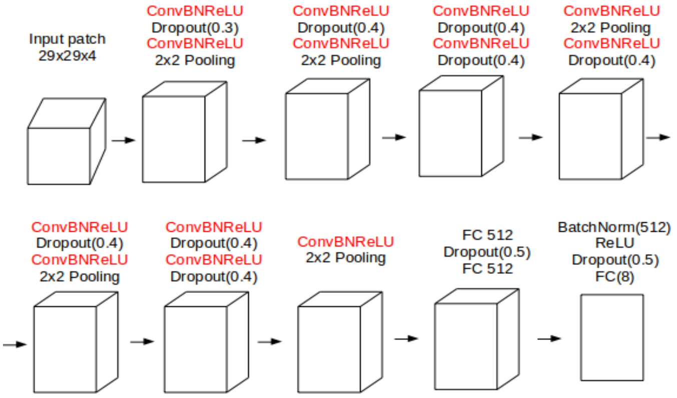


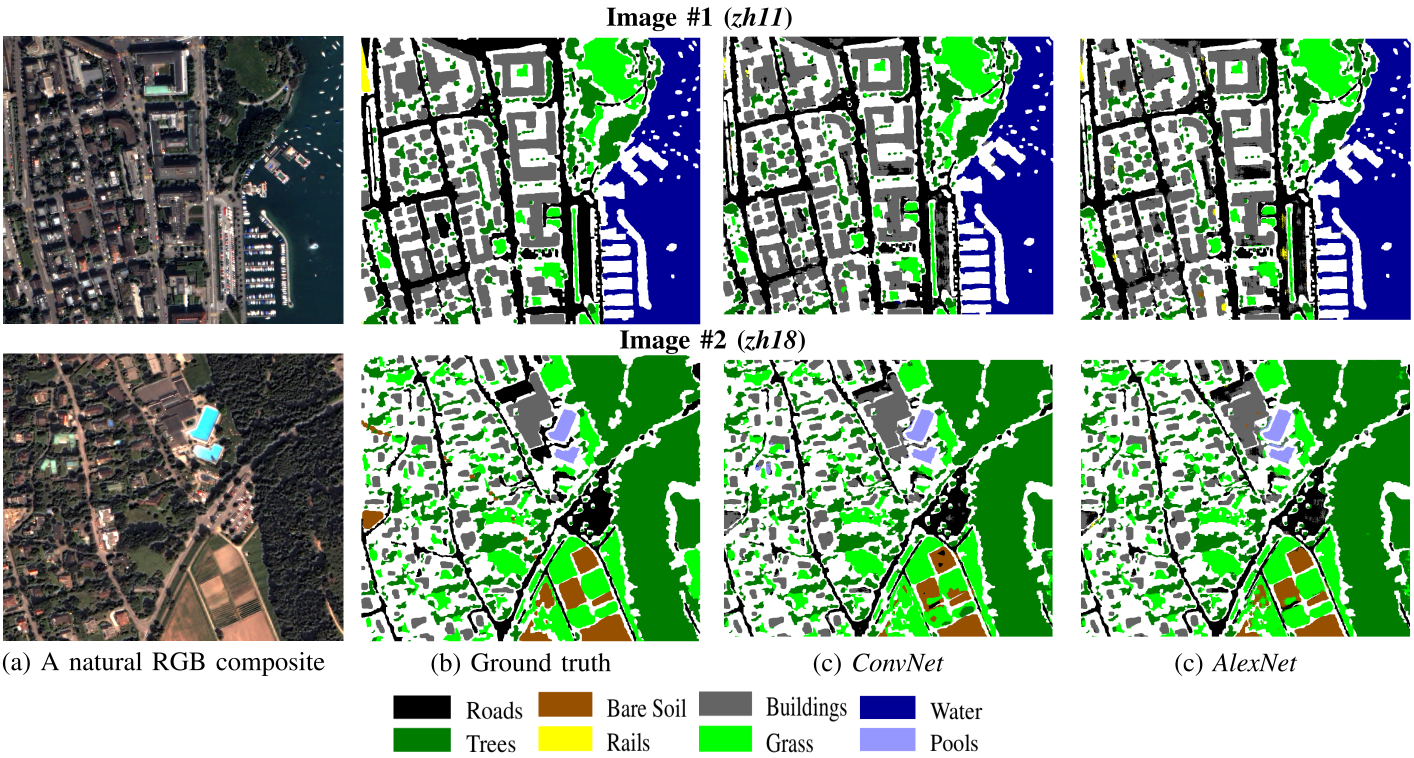
Fig. 3. Graphical illustration of *VGG* model. Function ConvBNRelu is depicted with red letters.

D. Implementation details

Regarding the implementation, all the models were trained with a learning rate of 1 for 36 epochs, while every 3 epochs the learning rate was reduced to half. The momentum was set to 0.9, the weight decay parameters to $5 \cdot 10^{-4}$ and the limit for the Threshold layer to 10^{-7}. The only exclusion was the *VGG* model which was trained for 40 epochs due to its greater depth. For the testing, *ConvNet* and *AlexNet* needed about 10-15 minutes while *VGG* needed about four hours on a GeForce GTX 980 GPU.

III. Experimental results and evaluation

For the evaluation of the tested architectures the publicly available Zurich Summer v1.0 dataset [12] was employed. In particular, the dataset contains 20 multispectral, very high resolution Quickbird images together with the ground truth for eight different classes. All VHR images were acquired over the city of Zurich in 2002. Every image has different dimensions



(a) A natural RGB composite (b) Ground truth (c) *ConvNet* (c) *AlexNet*

Fig. 4. The resulting semantic labeling maps after the application of the *ConvNet* (c) and *AlexNet* (d) deep learning models. Each class is presented with different color. The corresponding RGB image (a) and the ground truth data (b) are also shown.

of about 1000x1100 pixels, while the acquired Ground Sample Distance (GSD) is equal to 0.61 meters. Lastly, the different classes that are included in the dataset are the following eight: *Roads, Buildings, Trees, Grass, Bare Soil, Water, Railways* and *Swimming Pools*.

The training dataset was produced using 18 out of 20 images, while the remaining two were used for testing. The test images were selected so as to include all the eight different classes of the dataset. More analytically, Image #1 (*zh11* image from the dataset) includes the following classes: *Roads, Buildings, Trees, Grass, Water and Railways*, while, Image #2 (*zh18* image from the dataset) includes the following classes: *Roads, Buildings, Trees, Grass, Bare Soil and Swimming Pools*.

The training dataset was set up according to the following procedure: Firstly, the number of corresponding pixels was calculated for each category of an image. Ten percent of this number was randomly collected for each class and was used for extracting the patches. More specifically, patches of dimension 11x11, 21x21, 29x29, 33x33 and 45x45 were centred on every pixel under consideration. The final training dataset was a 4-dimensional vector of size N x 4 x F x F, where N is the number of patches, 4 is the number of available spectral channels, while F represents the patch dimensions that are produced at each time. The total number of training patches was approximately 1100000 for all the classes. Lastly, it should be mentioned that in all cases the training was performed from scratch and none data augmentation technique took place.

Different training and testing procedures were performed with the *ConvNet* model based on the different patch sizes. The resulting overall accuracy was examined for all sizes and the higher ones were used to train the two other deeper models *i.e., AlexNet* and *VGG*. In Figure 4 the resulting semantic labeling maps after the application of the *ConvNet* and *AlexNet* deep learning models are presented. After a close look one can observe that both models managed to compute adequate features in order to detect efficiently the different semantic labels.

For the quantitative evaluation, the overall accuracy (OA) and Kappa coefficient measures have been calculated. The resulting accuracy rates, when patches of different dimensions were considered, are shown in Table I for the case of the *ConvNet* model. Generally speaking, all tested patch dimensions resulted in high accuracy rates (>90%), while the Kappa coefficient rates exceed 87%. As expected, the lowest quantitative rates were reported for patches with relatively small dimensions *i.e.,* 11x11 pixels.

The highest accuracy rates were obtained for patch sizes of 29x29 and 45x45. In particular, the 29x29 resulted in slightly

Patch size	**OA %**		**Kappa coefficient**	
	Image #1	Image #2	Image #1	Image #2
11x11	90.3	91.1	0.872	0.886
21x21	92.4	92.7	0.900	0.890
29x29	94.8	**93.5**	0.911	**0.903**
33x33	93.2	92.3	0.910	0.886
45x45	**95.2**	92.4	**0.937**	0.886

TABLE I. THE RESULTING ACCURACY RATES FOR EXPERIMENTS WITH THE *ConvNet* MODEL AND PATCHES OF DIFFERENT SIZES. THE OVERALL ACCURACY (OA) AND KAPPA COEFFICIENT WERE OBTAINED FROM THE RESULTING CONFUSION MATRICES.

Models	AA %		Kappa coefficient	
	Image #1	Image #2	Image #1	Image #2
Learned RP 40m [12]	78.35		0.813	
ConvNet	84.7	**90.3**	0.911	0.903
AlexNet	**86.5**	89.5	0.910	0.897
VGG	80.9	89.9	**0.936**	**0.905**

TABLE II. THE RESULTING ACCURACY RATES OF THE THREE EMPLOYED DEEP LEARNING MODELS WHEN A 29X29 PATCH WAS CONSIDERED. RESULTS ARE COMPARED WITH THE METHOD IN [12].

higher mean OA (94.2%). The accuracy rates for the size of 33x33 were lower. Moreover, for a patch size of 45x45 even if it scored the highest rate for the case of Image #1 there was a failure to detect certain classes, like the *Railways*. The *Railways* class resulted into a zero accuracy in this particular case.

For a patch size of 29x29 we did comparisons with the other two deeper models *i.e.,* the *AlexNet* and *VGG*. These results are presented in Table II. The highest mean average accuracy (AA) rates were obtained from the *AlexNet* and *ConvNet* models. The *VGG* network scored almost the same with the other two methods, while it scored higher Kappa coefficient. It should be noted that all deep learning models outperformed the mean average accuracy and the Kappa coefficient of the method *'Learned RP 40m'* in [12] by at least 2% on the same dataset. This implies that deep convolutional architectures seem to construct more adequate and robust features, while the computations at the deeper layers with a relatively small patch size do not significantly impede the mean average accuracy result.

For a more detailed quantitative evaluation, the highest accuracy rates obtained for each land cover class were also compared in Table III. Apart from the class *Trees* for which the *'Passive RP'* method in [12] resulted in the highest accuracy rates, in all other cases the CNN architectures outperformed the methods considered in [12]. Moreover, the deeper *VGG* model was the one that resulted in the highest accuracy rates for the majority of the land cover classes, even though it did not have the highest mean average accuracy. This is mainly due to its quite low accuracy on the *Railways* class.

IV. CONCLUSION

In this paper, the performance of different deep-learning architectures under a patch-based framework for the semantic labeling of urban scenes was evaluated. Different patch sizes were employed and tested under the *ConvNet* model. The *AlexNet* and *VGG* models were also employed for training and testing on the publicly available, multispectral, very high resolution Summer Zurich v1.0 dataset. The overall results demonstrate the robustness of the high level features that are computed with the employed deep architectures for urban scene semantic labeling with very high resolution satellite data. In particular, all implemented deep learning models outperformed the mean average accuracy of the method *'Learned RP 40m'* in [12] by at least 2% on the same dataset.

Classes			Image #1		Image #2	
	Acc.(%)	Method [12]	Acc.(%)	Method	Acc.(%)	Method
Roads	84.02	*Learned RP 40M*	**93.24**	*VGG*	84.62	*VGG*
Buildings	87.04	*RF Unary*	93.24	***ConvNet***	**95.14**	*VGG*
Trees	**95.10**	*Passive RP*	93.46	*VGG*	92.72	*VGG*
Grass	86.92	*Learned RP 40m*	**96.46**	*VGG*	92.72	*VGG*
Soil	75.51	*Learned MRP*	-	-	**76.36**	***ConvNet***
Water	94.31	*Learned MRP*	**99.76**	*VGG*	-	-
Rails	21.35	*Learned RP 40m*	**50.89**	*AlexNet*	-	-
Pools	92.13	*Learned MRP*	-	-	**99.77**	*VGG*

TABLE III. CLASS LEVEL COMPARISON BETWEEN THE DIFFERENT EMPLOYED MODELS AND METHOD PROPOSED IN [12].

REFERENCES

[1] W. Liao, R. Bellens, A. Pizurica, W. Philips, and Y. Pi, "Classification of hyperspectral data over urban areas using directional morphological profiles and semi-supervised feature extraction," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 4, pp. 1177–1190, Aug 2012.

[2] N. Longbotham, C. Chaapel, L. Bleiler, C. Padwick, W. J. Emery, and F. Pacifici, "Very high resolution multiangle urban classification analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 4, pp. 1155–1170, April 2012.

[3] X. Zhang, S. Du, and Y. C. Wang, "Semantic classification of heterogeneous urban scenes using intrascene feature similarity and interscene semantic dependency," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 5, pp. 2005–2014, May 2015.

[4] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 487–495.

[5] W. Anran, C. Jianfei, L. Jiwen, and C. Tat-Jen, "Modality and component aware feature fusion for rgb-d scene classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[6] K. Nogueira, O. A. B. Penatti, and J. A. dos Santos, "Towards better exploiting convolutional neural networks for remote sensing scene classification," *CoRR*, vol. abs/1602.01517, 2016. [Online]. Available: http://arxiv.org/abs/1602.01517

[7] M. Papadomanolaki, M. Vakalopoulou, S. Zagoruyko, and K. Karantzalos, "Benchmarking deep learning frameworks for the classification of very high resolution satellite multispectral data," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. III-7, pp. 83–88, 2016. [Online]. Available: http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/III-7/83/2016/

[8] K. Makantasis, K. Karantzalos, A. Doulamis, and N. Doulamis, "Deep supervised learning for hyperspectral data classification through convolutional neural networks," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2015, pp. 4959–4962.

[9] M. Vakalopoulou, K. Karantzalos, N. Komodakis, and N. Paragios, "Building detection in very high resolution multispectral data with deep learning features," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2015, pp. 1873–1876.

[10] M. Volpi and D. Tuia, "Dense semantic labeling of sub-decimeter resolution images with convolutional neural networks," *ArXiv e-prints*, Aug. 2016.

[11] S. Paisitkriangkrai, J. Sherrah, P. Janney, and A. V.-D. Hengel, "Effective semantic pixel labelling with convolutional networks and conditional random fields," in *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2015, pp. 36–43.

[12] M. Volpi and V. Ferrari, "Semantic segmentation of urban scenes by learning local class interactions," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2015.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.