BENCHMARKING DEEP LEARNING FRAMEWORKS FOR THE CLASSIFICATION OF VERY HIGH RESOLUTION SATELLITE MULTISPECTRAL DATA

M. Papadomanolaki[a*], M. Vakalopoulou[a*], S. Zagoruyko[b], K. Karantzalos[a]

[a] Remote Sensing Laboratory, National Technical University,
Zographou campus, 15780, Athens, Greece
mar.papadomanolaki@gmail.com; *mariavak@central.ntua.gr*; *karank@central.ntua.gr*

[b] Imagine/Ligm, Ecole des Ponts ParisTech,
Cite Descartes, 77455 Champs-sur-Marne, France
sergey.zagoruyko@imagine.enpc.fr
sergey.zagoruyko@imagine.enpc.fr

Commission VII, WG IV

KEY WORDS: Machine Learning, Classification, Land Cover, Land Use, Convolutional, Neural Networks, Data Mining

ABSTRACT:

In this paper we evaluated deep-learning frameworks based on Convolutional Neural Networks for the accurate classification of multispectral remote sensing data. Certain state-of-the-art models have been tested on the publicly available *SAT-4* and *SAT-6* high resolution satellite multispectral datasets. In particular, the performed benchmark included the *AlexNet*, *AlexNet-small* and *VGG* models which had been trained and applied to both datasets exploiting all the available spectral information. Deep Belief Networks, Autoencoders and other semi-supervised frameworks have been, also, compared. The high level features that were calculated from the tested models managed to classify the different land cover classes with significantly high accuracy rates *i.e.,* above 99.9%. The experimental results demonstrate the great potentials of advanced deep-learning frameworks for the supervised classification of high resolution multispectral remote sensing data.

1. INTRODUCTION

The detection and recognition of different objects and land cover classes from satellite imagery is a well studied problem in the remote sensing community. The numerous national and commercial earth observation programs continuously provide data with different spatial, spectral and temporal characteristics. In order to operationally exploit these massive streams of imagery, advanced processing, mining and recognition tools are required. These tools should be able to timely extract valuable information regarding the various terrain objects and land cover, land use status.

How automated and how accurate these recognition tools are, is the critical aspect for their applicability and operational use. Regarding automation, although unsupervised and semi-supervised approaches possess a native advantage, when comes to big data from space with important spatial, spectral and temporal variability, efficient generic tools may be based on supervised approaches which have been trained to handle and classify such datasets [Karantzalos et al., 2015, Cavallaro et al., 2015].

Among supervised classification approaches, Support Vector Machines (SVM) [Vapnik, 1998] and Random Forests [Breiman, 2001] have been broadly used for remote sensing applications [Camps-Valls and Bruzzone, 2009, Tokarczyk et al., 2015]. Under supervised frameworks every training dataset must be adequately exploited in order to describe compactly and adequately each class. To this end, training datasets may consist of a combination of spectral bands, morphological filters [Lefevre et al., 2007], texture [Volpi et al., 2013], point descriptors [Wang et al., 2013], gradient orientation [Benedek et al., 2012], *etc*.

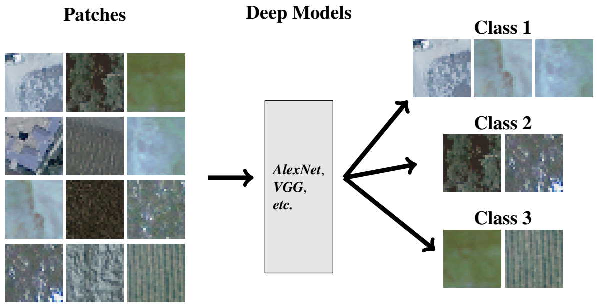


Figure 1: Certain state-of-the-art models have been tested on the publicly available *SAT-4* and *SAT-6* high resolution satellite multispectral datasets. All the models have been designed and trained based on the *DeepSat* dataset.

Recently, deep learning architectures have gained attention in computer vision and remote sensing by delivering state-of-the-art results on image classification [Sermanet et al., 2013], [Krizhevsky et al., 2012], object detection [LeCun et al., 2004] and speech recognition [Xue et al., 2014]. Several deep architectures [Deng, 2014, Schmidhuber, 2015] have been employed, with the Deep Belief Networks, Autoencoders, Convolutional Neural Networks and Deep Boltzmann Machines being some of the most commonly used in the literature for a variety of problems. In particular, for the classification of remote sensing data certain deep architectures have provided highly accurate results [Mnih and Hinton, 2010, Chen et al., 2014, Vakalopoulou et al., 2015, Basu et al., 2015, Makantasis et al., 2015, Marmanis et al., 2016].

Deep architectures require a significant amount of training data, while labelled remote sensing data are not broadly available. Recently, a new publicly available dataset with a large number of

[*] These authors contributed equally to this work.

doi:10.5194/isprsannals-III-7-83-2016

training data was released [Basu et al., 2015]. *DeepSat* contains patches extracted from the National Agriculture Imagery Program (NAIP) dataset with about 330,000 scenes spanning the entire Continental United States and approximately 65 terabytes of size. The images consist of 4 bands: red, green, blue and Near Infrared (NIR) and were acquired at a ground sample distance (GSD) of 1 meter, having horizontal accuracy up to 6 meters. The dataset is composed of two discrete units, each of them having two different group of patches: one for training and one for testing. *SAT-4* is the first unit and it has four different classes which are: barren land, trees, grassland and a class that consists of all land cover classes other than the above three. It contains 400.000 training and 100.000 testing patches. The second unit, *SAT-6*, contains six different classes: barren land, trees, grassland, roads, buildings and water bodies. It contains 324.000 training and 81.000 testing patches. The large number of labelled data that *DeepSat* provides, makes it ideal for benchmarking different deep architectures.

In this paper, motivated by the recent advances on deep convolutional neural networks, we benchmark the performance of certain models for classifying multispectral remote sensing data (Figure 1). Based on the *DeepSat* dataset, we have trained different models and reported on their performances. In particular, *AlexNet*, *AlexNet-small* and *VGG* models have been implemented and trained on the *DeepSat* dataset [Krizhevsky et al., 2012, Jaderberg et al., 2015, Ioffe and Szegedy, 2015, Vakalopoulou et al., 2015]. The high accuracy rates demonstrate the potentials of advanced deep-learning frameworks for the supervised classification of high resolution multispectral remote sensing imagery. Comparing with Deep Belief Networks, Autoencoders and Semi-supervised frameworks [Basu et al., 2015] the proposed here *AlexNet* and *VGG* deep architectures outperform the state-of-the-art delivering classification accuracy rates above 99.9%.

The remainder of the paper is organized as follows. In Section 2., we briefly describe different deep learning models while in Section 3. we present and discuss their results. The last section provides a short summary of the contributions and examines potential future directions.

2. DEEP-LEARNING FRAMEWORKS

In this section all the tested models and their parameters are presented. Both training and testing datasets had been normalised before inserted into the networks. The implementation of all compared deep learning frameworks was performed in the open source Torch deep learning library [Collobert et al., 2011], while the specific implementation is available in authors' *GitHub* accounts.

2.1 *AlexNet-Pretrained* Network

Similar to [Vakalopoulou et al., 2015, Marmanis et al., 2016] the already pretrained *AlexNet* network [Krizhevsky et al., 2012] has been employed, here, for feature extraction. In particular, features from the last layer (FC7) were extracted using two spectral band combinations (red-green-blue and NIR-red-green). In this way, a vector with high level features of size 2x4096 has been created for each patch. Using the training dataset an SVM classifier has been trained (Figure 2) and then the produced model was used for the classification of the testing patches.

The main drawback of this specific setup is the high dimensionality of the employed feature vector (*i.e.,* 2 x 4096) as the pretrained model can not handle more than three spectral bands per patch.

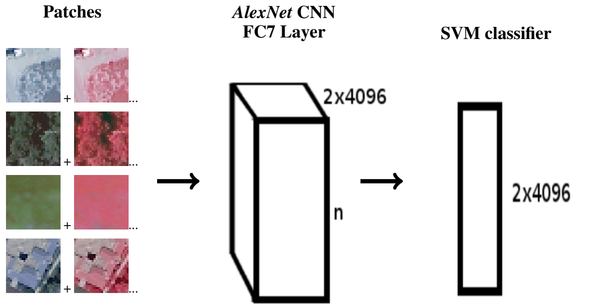


Figure 2: The FC7 layer of the Pretrained *AlexNet* network has been employed for extracting features and train a SVM classifier. Two different band combinations (red-green-bue and NIR-red-green) have been formulated in order to exploit all the available spectral information.

2.2 *AlexNet* Network

In order to overcome the previous problem we trained an *AlexNet* Network using the *DeepSat* dataset. The model consists of 22 layers: 5 convolutional, 3 pooling, 6 transfer functions, 3 fully connected and 5 dropout and threshold. The model follows the patterns as depicted in Figure 3. More specifically, the first convolutional layer receives the raw input patch which consists of 4 channels (or input planes) and is of size 28x28. The image is filtered with kernels of size 4x3x3 and a stride of 1 pixel, producing an output volume of size 16x26x26. The second layer is a transfer function one which applies the rectified linear unit (ReLU) function element-wise to the input tensor. Thus, the dimensions of the image remain unchanged. Next comes a max pooling layer, which is used to progressively reduce the spatial size of the image in order to restrict the amount of network computation and parameters and protect from overfitting. This pooling layer uses kernels of size 2 and a stride of 2, producing an output volume of size 16x13x13.

The next 3 layers follow the same pattern (Convolutional-ReLU-MaxPooling). The Convolutional layer accepts the 16x13x13 volume and produces an output of size 48x13x13 by using 3x3 kernels, with a stride of 1 and a zero padding of 1 pixel. The Convolutional layer is followed by a ReLU and a MaxPooling layer. The latter having kernels of size 3 and a stride of 2 is delivering output volumes of size 48x6x6. The seventh layer is also a Convolutional layer which delivers an output of size 96x6x6 by applying 3x3 kernels and stride and zero padding of 1. The eighth layer is a ReLU one. Layers 9,10 and 11,12 follow the same pattern (Convolutional-ReLU). The ninth layer is filtering the input volume with kernels of size 3x3, with a stride of 1 and zero padding of 1, delivering an output of size 64x6x6. The eleventh convolutional layer uses the same hyperparameters. The twelfth convolutional layer is a maxpooling one, which uses kernels of size 2 and a stride of 2 to produce an output of size 64x3x3.

After that, we use some fully-connected (FC) layers. The thirteenth layer is a simple View one which converts the given volume of size 64x3x3 to an output volume of size 64*3*3x1=576x1. Next comes a Dropout layer with a probability of 0.5, which masks part of the current input using binary functions from a Bernoulli distribution. This layer sets to zero the output of each hidden neuron with probability of 0.5. The fifteenth layer is a simple linear one, which converts the input volume of size 576x1 to an output volume of size 200x1. The linear layer is followed by a Threshold (TH) one. The next 3 layers are of the same logic (Dropout-Linear-Threshold) and have the same hyperparameters. The 21-th layer is a Linear one, which results to an output volume

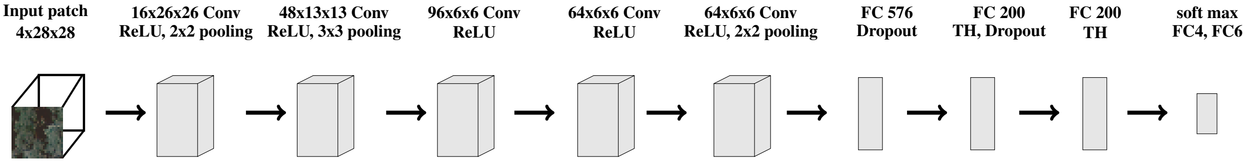


Figure 3: A brief illustration of the *AlexNet* model which was employed here. The network takes as input patches of 4x28x28 (dimensions). The network consists of 5 convolutional, 3 fully connected layers and 6 transfer function. A 4 or 6 way soft-max layer is applied depending on the dataset.

of size 4x1, where 4 is the number of class scores. Lastly, we use a soft-max layer with 4 or 6 ways depending on the tested dataset.

Regarding the implementation, we trained the model with a learning rate of 1 for 36 epochs, while every 3 epochs the learning rate was reduced at half. We set the momentum to 0.9, the weight decay parameters to 0.0005 and the limit for the Threshold layer to 0.000001.

2.3 *AlexNet-small* Network

Another model that we tested was a simpler small *AlexNet* Network. The model consists of 10 layers. The first layer is a convolutional layer and is feeded with the original input image of size 4x28x28, producing a volume of size 32x24x24. In this setup, we did not use the ReLU function but the Tangent one. Consequently the Tangent layer comes after the first convolutional one, applying the tangent function element-wise to the input tensor. After that, follows a third max pooling layer which narrows down the size of the image from 32x24x24 to 32x8x8. The next 3 layers follow the same pattern and result to an output volume of size 64x2x2. Finally, 4 fully-connected layers follow. At this point we should mention that the model does not contain Dropout layers contrary to the previous full *AlexNet* model. Nonetheless, results were satisfactory, mostly because of the 2 max pooling layers which made the spatial size of the images smaller and controled overfitting.

Finally, the parametrization for the *AlexNet-small* was chosen to be similar with the previous full *AlexNet* one. Again we trained the model with a learning rate of 1 for 36 epochs. The learning rate was reduced in half at every 3 epochs. We set the momentum to 0.9 and the weight decay parameters to 0.0005.

2.4 *VGG* Network

Moreover, we experimented with the recent *VGG* model [Jaderberg et al., 2015] which was initially proposed for text recognition. The model consists of 59 levels and it repeatedly makes use of specific layers that include dropout and batch normalization [Ioffe and Szegedy, 2015]. It should be noted that these two kinds of layers are very important in order to accelerate the entire process and avoid overfitting, as the parameters of each training layer continuously change. The first group of layers has 4 levels and they have the following pattern: the training model starts with a convolutional layer that takes the input image of size 4x28x28 and it produces an output of size 64x28x28. Then, a batch normalization layer is applied with a value of 0.001 added to the standard deviation of the input maps. After that, a ReLU layer is implemented and lastly, a dropout layer with a probability of 0.3. The next group of layers has also 4 levels and it follows the same logic, except the last layer, which is a max pooling one of kernel size 2 and a stride of 2 that converts the input from 62x28x28 to 64x14x14. These two groups of layers are repeatedly used with the same hyperparameters, except dropout which sometimes has a probability of 0.4. The last 7 layers of the entire training model include some fully connected layers.

The deeper architecture of the *VGG* model requires bigger amount of data as well as more training time. For that reason data augmentation with horizontal and vertical flips was performed. Finally, the network was trained for 100 epochs, reducing the learning rate in half every 10 epochs.

2.5 Deep Belief Networks, Autoencoders, Semi-supervised frameworks

Last but not least, the aforementioned CNN-based networks were compared with classification frameworks which have been recently evaluated in [Basu et al., 2015] for the *DeepSat* dataset. In particular, approaches based on Deep Belief Networks, Stacked Denoising Autoencoder and a semi-supervised one were employed. The training and testing included both datasets using different parameters, features maps and configurations for each technique. [Basu et al., 2015] concluded that the semi-supervised approach was more suitable for the *DeepSat* dataset performing 97.95% and 93.92% accuracies for the *SAT-4* and *SAT-6* datasets respectively.

3. EXPERIMENTAL RESULTS AND EVALUATION

In this section the performed experimental results are presented along with the comparative study. Both training and testing have been performed separately for the two datasets of *SAT-4* and *SAT-6*. For the quantitative evaluation, the accuracy and precision

Benchmarking on *DeepSat SAT-4* dataset

	***AlexNet* Pretrained**		***AlexNet***		***AlexNet-small***		***VGG***	
LC Class	Accuracy	Precision	Accuracy	Precision	Accuracy	Precision	Accuracy	Precision
Barren Land	99.24%	99.02%	99.96%	99.88%	99.83%	99.54%	99.96%	99.85%
Trees	99.73%	99.51%	99.99%	99.98%	99.95%	99.90%	99.99%	99.97%
Grassland	99.13%	96.76%	99.96%	99.90%	99.81%	99.59%	99.95%	99.99%
Other	99.77%	99.73%	99.98%	99.98%	99.96%	99.95%	99.99%	99.99%
Overall	99.46%	98.75%	**99.98**%	99.94%	99.86%	99.75%	**99.98%**	**99.95%**

Table 1: Resulting classification accuracy and precision rates after the application of different deep architectures in the ***SAT-4*** dataset.

Benchmarking on *DeepSat SAT-6* dataset

	***AlexNet* Pretrained**		***AlexNet***		***AlexNet-small***		***VGG***	
LC Class	Accuracy	Precision	Accuracy	Precision	Accuracy	Precision	Accuracy	Precision
Barren Land	99.91%	98.72%	99.95%	99.41%	99.96%	99.70%	99.99%	100%
Trees	99.14%	98.78%	99.86%	99.58%	99.77%	99.33%	99.96%	99.87%
Grassland	99.04%	99.72%	99.97%	99.92%	99.95%	99.84%	99.99%	99.96%
Roads	99.16%	96.62%	99.84%	99.61%	99.74%	99.43%	99.89%	99.95%
Buildings	99.92%	98.93%	99.95%	99.08%	99.96%	99.08%	99.99%	99.61%
Water Bodies	100%	100%	100%	100%	99.99%	100%	100%	100%
Overall	99.57%	98.80%	99.93%	99.60%	99.90%	99.56%	**99.98**%	**99.91**%

Table 2: Resulting classification accuracy and precision rates after the application of different deep architectures in the ***SAT-6*** dataset.

measures have been calculated.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

where TP is the number of correctly classified patches, TN is the number of the patches that do not belong to the specific class and they were not classified correctly. FN is the number of patches that belong to the specific class but weren't correctly classified and FP is the number of patches that do not belong to the specific class but have been wrongly classified.

Regarding the experiments performed on the *SAT-4* dataset, the calculated precision and accuracy after the application of the different deep learning frameworks are presented in Table 1. One can observe that all employed models have obtained quite highly accurate results. More specifically, the overall accuracy rates were in all cases more than 99.4%, while the estimated precision was more than 98.7%. Therefore, the employed deep architectures can address the classification task in this particular dataset quite successfully. As expected, the estimated accuracy and precision rates for the Pretrained *AlexNet* were lower than the other models, since it was trained on the ImageNet dataset. Moreover, the use of the AlexNet pretrained network makes the computational complexity much higher than in the other models.

The other three models successfully classified all the corresponding classes, achieving accuracy rates more than 99.90%. Additionally, as expected the *AlexNet* network performs slightly better than the *AlexNet-small*, which means that the ReLU layer and deeper architectures are more suitable for this specific dataset. In particular, the class that scored lower regarding the accuracy and precision was the grassland, as some patches were misclassified as barren land or trees. The *VGG* and the *AlexNet* models resulted into the higher accuracy/precision rates (*i.e.,* above 99.94%).

Method	**Overall Accuracy (%)** *SAT-4*	*SAT-6*
DBN *[Basu et al., 2015]*	81.78	76.41
CNN *[Basu et al., 2015]*	86.83	79.06
SDAE *[Basu et al., 2015]*	79.98	78.43
Semi-supervised *[Basu et al., 2015]*	97.95	93.92
Pretrained-AlexNet *[Vakalopoulou et al., 2015]*	99.46	99.57
AlexNet *(Proposed)*	**99.98**	99.93
AlexNet-small *(Proposed)*	99.86	99.90
VGG *(Proposed)*	**99.98**	**99.98**

Table 3: The resulting overall accuracy rates after the application of different learning frameworks for both datasets.

Regarding the experiments performed with the *SAT-6* dataset, results from the quantitative evaluation are presented in Table 2. As in *SAT-4*, the pretrained *AlexNet* model resulted into the lowest accuracy and precision rates comparing to the other ones. The *VGG* model was the one that resulted into slightly higher accuracy rates than the *AlexNet* and *AlexNet-small* models. In all cases the overall accuracy rates were higher than 99.6%. Moreover, one can observe that the *water bodies* class was easy to be discriminated from the other ones, as in all cases the calculated accuracy was more than 99.99%. On the other hand, the *Roads* class was the one that resulted in the lowest accuracy rates as certain misclassification cases occurred with the *Trees* and *Grassland* classes.

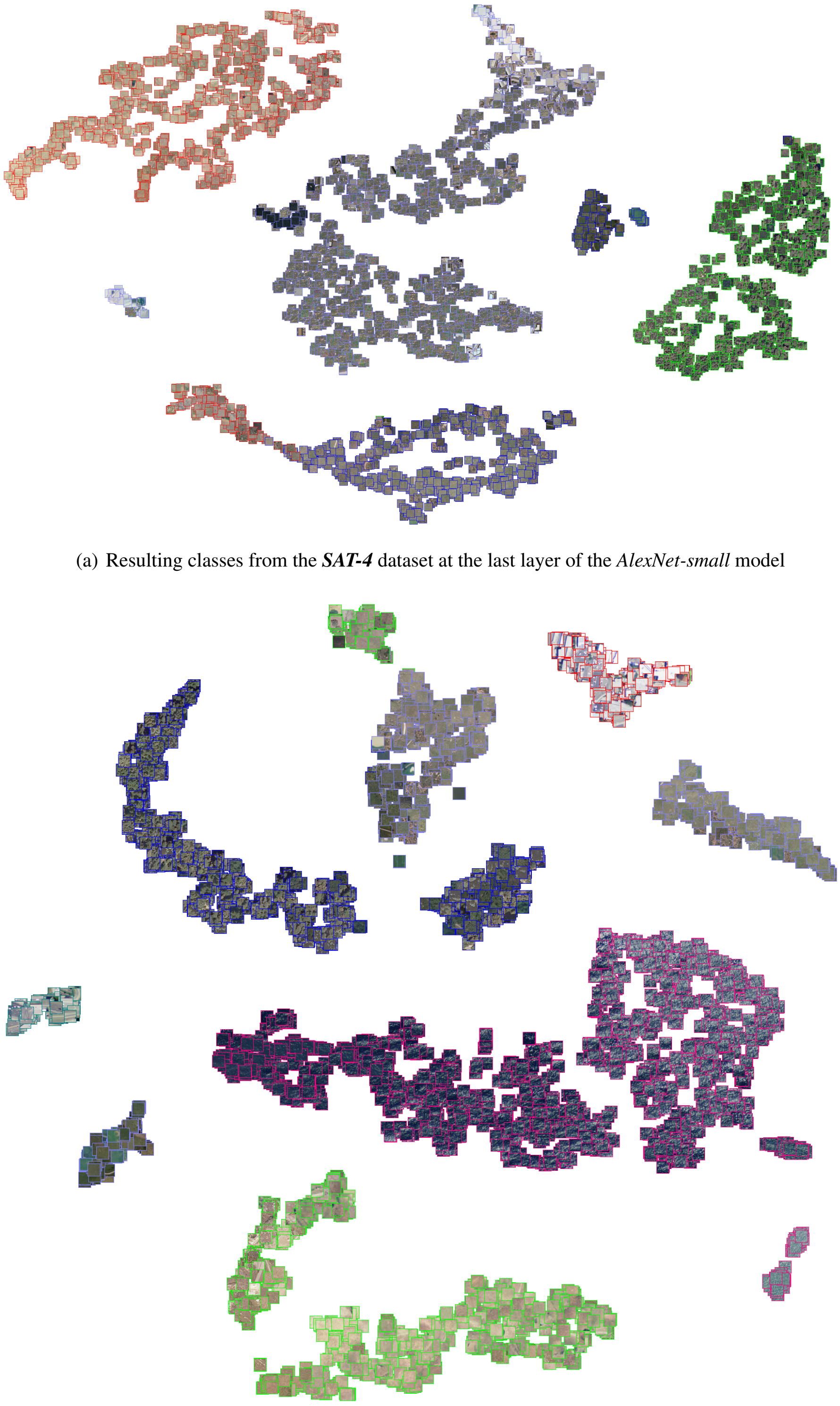
In addition, for the qualitative evaluation of the performed experiments the *t-SNE* technique [Van Der Maaten, 2014] was employed. *t-SNE* has been tested in different computer vision datasets *e.g.,* MNIST, NIPS dataset, *etc.* and it is suitable for the visualization of high-dimensional large-real world datasets. In Figure 4 the visualization of the last layer features of the *AlexNet-small* model for both *SAT-4* and *SAT-6* is shown. In particular, different classes are represented with different colours in the borders of the patches. One can observe that the different classes are well separated in space, which can justify the high accuracy rates that have been delivered and reported quantitatively in Table 1 and Table 2.

Last but not least, the deep architectures of *AlexNet & VGG* were compared with the recently proposed and evaluated ones from [Basu et al., 2015]. In Table 3, the overall accuracy rates for both *SAT-4* and *SAT-6* datasets are presented. This benchmark included results after the application of Deep Belief Network (DBN), Convolutional Neural Network (CNN), Stacked Denoising Autoencoder (SDAE), a semi-supervised learning framework, *AlexNet* (pre-trained and small) and *VGG*. In [Basu et al., 2015], the highest accuracy rates were obtained from the semi-supervised classification framework and were 97.95% and 93.9% for the *SAT-4* and *SAT-6*, respectively. As one can observe all the pro-

Evaluating results from *VGG* without the NIR band

	SAT-4		***SAT-6***	
LC Class	Accuracy	Precision	Accuracy	Precision
Barren Land	99.96%	98.85%	99.99%	99.89%
Trees	99.99%	99.97%	99.90%	99.64%
Grassland	99.95%	99.99%	99.97%	99.85%
Others	99.99%	99.99%	-	-
Roads	-	-	99.99%	99.95%
Buildings	-	-	99.99%	99.95%
Water Bodies	-	-	100%	100%
Overall	99.98%	99.95%	99.96%	99.88%

Table 4: Resulting classification accuracy and precision rates after the application of the *VGG* model at both datasets without the NIR Band.



(a) Resulting classes from the ***SAT-4*** dataset at the last layer of the *AlexNet-small* model

(b) Resulting classes from the ***SAT-6*** dataset at the last layer of the *AlexNet-small* model

Figure 4: Qualitative evaluation based on the *t-SNE* technique. Results from the *SAT-4* (top) and the *SAT-6* (bottom) datasets are presented. Different classes are defined with different colours in the borders of the patches. This visualisation corresponds to the last layer of the *AlexNet-small* model.

posed and applied models in this paper (including the pre-trained *AlexNet* which scored lower that the other ones) outperformed the semi-supervised framework of [Basu et al., 2015]. The proposed deep models efficiently exploited the available spectral information (all available spectral bands) and created deep features that could accurately discriminate the different classes.

In order to evaluate the contribution of the NIR band, we moreover experimented with training the models with and without its use. In particular, experimental results after applying the VGG model to both *SAT-4* and *SAT-6* dataset are presented in Table 4. As expected the classes of *Tress* and *Grassland* present lower precision and accuracy rates when the NIR band was excluded from the training process. However, the overall accuracy and precision rates were slightly different, especially on the *SAT-6* dataset.

4. CONCLUSIONS

In this paper, experimental results after benchmarking different deep-learning frameworks for the classification of high resolution multispectral data were presented. Deep Belief Network (DBN), Convolutional Neural Network (CNN), Stacked Denoising Autoencoder (SDAE), a semi-supervised learning framework, *AlexNet* (pre-trained and small) and *VGG* were among the frameworks that were evaluated. The evaluation was based on the publicly available *DeepSat* dataset including both *SAT-4* and *SAT-6*. Comparing with Deep Belief Networks, Autoencoders and Semi- supervised frameworks [Basu et al., 2015] the proposed here *AlexNet* and *VGG* deep architectures outperform the state-of-the-art delivering high classification accuracy rates above 99.9%. The quite promising quantitative evaluation indicates the high potentials of deep architectures towards the design of operational remote sensing classification tools.

REFERENCES

Basu, S., Ganguly, S., Mukhopadhyay, S., DiBiano, R., Karki, M. and Nemani, R., 2015. DeepSat - A Learning framework for Satellite Imagery. In: ACM SIGSPATIAL.

Benedek, C., Descombes, X. and Zerubia, J., 2012. Building development monitoring in multitemporal remotely sensed image pairs with stochastic birth-death dynamics. Pattern Analysis and Machine Intelligence, IEEE Transactions on 34(1), pp. 33–50.

Breiman, L., 2001. Random forests. Machine Learning 45(1), pp. 5–32.

Camps-Valls, G. and Bruzzone, L., 2009. Kernel Methods for Remote Sensing Data Analysis. Wiley.

Cavallaro, G., Riedel, M., Richerzhagen, M., Benediktsson, J. and Plaza, A., 2015. On understanding big data impacts in remotely sensed image classification using support vector machine methods. Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of PP(99), pp. 1–13.

Chen, Y., Lin, Z., Zhao, X., Wang, G. and Gu, Y., 2014. Deep learning-based classification of hyperspectral data. Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of 7(6), pp. 2094–2107.

Collobert, R., Kavukcuoglu, K. and Farabet, C., 2011. Torch7: A Matlab-like Environment for Machine Learning. In: BigLearn, NIPS Workshop.

Deng, L., 2014. A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Transactions on Signal and Information Processing.

Ioffe, S. and Szegedy, C., 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: D. Blei and F. Bach (eds), Proceedings of the 32nd International Conference on Machine Learning (ICML-15), pp. 448–456.

Jaderberg, M., Simonyan, K., Vedaldi, A. and Zisserman, A., 2015. Deep Structured Output Learning for Unconstrained Text Recognition. In: International Conference on Learning Representations.

Karantzalos, K., Bliziotis, D. and Karmas, A., 2015. A scalable geospatial web service for near real-time, high-resolution land cover mapping. Selected Topics in Applied Earth Observations and Remote Sensing, IEEE Journal of PP(99), pp. 1–10.

Krizhevsky, A., Sutskever, I. and Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. In: F. Pereira, C. Burges, L. Bottou and K. Weinberger (eds), Advances in Neural Information Processing Systems 25, Curran Associates, Inc., pp. 1097–1105.

LeCun, Y., Huang, F. J. and Bottou, L., 2004. Learning methods for generic object recognition with invariance to pose and lighting. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, Vol. 2, pp. II–97–104 Vol.2.

Lefevre, S., Weber, J. and Sheeren, D., 2007. Automatic building extraction in vhr images using advanced morphological operators. In: Urban Remote Sensing Joint Event, 2007, pp. 1–5.

Makantasis, K., Karantzalos, K., Doulamis, A. and Doulamis, N., 2015. Deep supervised learning for hyperspectral data classification through convolutional neural networks. In: Geoscience and Remote Sensing Symposium (IGARSS), 2015 IEEE International, pp. 4959–4962.

Marmanis, D., Datcu, M., Esch, T. and Stilla, U., 2016. Deep learning earth observation classification using imagenet pretrained networks. IEEE Geoscience and Remote Sensing Letters 13(1), pp. 105–109.

Mnih, V. and Hinton, G., 2010. Learning to detect roads in high-resolution aerial images. In: K. Daniilidis, P. Maragos and N. Paragios (eds), Computer Vision ECCV 2010, Lecture Notes in Computer Science, Vol. 6316, Springer Berlin Heidelberg, pp. 210–223.

Schmidhuber, J., 2015. Deep learning in neural networks: An overview. Neural Networks 61, pp. 85 – 117.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. and LeCun, Y., 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. CoRR.

Tokarczyk, P., Wegner, J., Walk, S. and Schindler, K., 2015. Features, color spaces, and boosting: New insights on semantic classification of remote sensing images. Geoscience and Remote Sensing, IEEE Transactions on 53(1), pp. 280–295.

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

Van Der Maaten, L., 2014. Accelerating t-SNE Using Tree-based Algorithms. Journal of Machine Learning Research 15(1), pp. 3221–3245.

Vapnik, V. N., 1998. Statistical Learning Theory. Wiley-Interscience.

Volpi, M., Tuia, D., Bovolo, F., Kanevski, M. and Bruzzone, L., 2013. Supervised change detection in VHR images using contextual information and support vector machines. International Journal of Applied Earth Observation and Geoinformation 20(0), pp. 77 – 85.

Wang, M., Yuan, S. and Pan, J., 2013. Building detection in high resolution satellite urban image using segmentation, corner detection combined with adaptive windowed hough transform. In: Geoscience and Remote Sensing Symposium (IGARSS), 2013 IEEE International, pp. 508–511.

Xue, S., Abdel-Hamid, O., Jiang, H., Dai, L. and Liu, Q., 2014. Fast adaptation of deep neural network based on discriminant codes for speech recognition. Audio, Speech, and Language Processing, IEEE/ACM Transactions on 22(12), pp. 1713–1725.