Deep Learning Architecture for Forest Detection in Satellite Data

Gabriel D. Caffaratti[1,3], Martin G. Marchetta[1], Raymundo Q. Forradellas[1], Leonardo D. Euillades[2,3], and Pablo A. Euillades[2,3]

[1] Intelligent Systems Laboratory (LABSIN)
[2] Special Training and Computer Assisted Engineering Development Institute (CEDIAC)
School of Engineering - National University of Cuyo
Centro Universitario, Mendoza, Argentina
[3] CONICET
{gabriel.caffaratti, martin.marchetta, leonardo.euillades pablo.euillades}@ingenieria.uncuyo.edu.ar
kike@uncu.edu.ar

Abstract. Deep Learning algorithms have achieved great progress in different applications due to their training capabilities, parameter reduction and increased accuracy. Image processing is a particular area that has received recent attention promoted by the growing processing power and data availability. Remote sensing devices provide image-like data that can be used to characterize Earth's natural or artificial phenomena. Particularly, forest detection is important in many applications like flooding simulations, analysis of forest health or detection of area desertification. The existing techniques for forest detection based on satellite data lack accuracy or still require human expert intervention to correct recognition errors or parameter setup. In this work a Deep Learning architecture for forest detection is presented, that aims at increasing accuracy and reducing expert dependency. A data preprocessing procedure, analysis and dataset composition for robust automatic forest detection is described. The proposed approach was validated with real SRTM and Landsat-8 satellite data.

Keywords: Remote sensing, Forest detection, Deep learning

1 Introduction

The characteristics of the land surface, if adequately processed, can provide fundamental information for many applications like modeling and simulating atmospheric, hydrologic and ecological processes occurring on the surface of the Earth [17]. Recently, some projects like the Shuttle Radar Topography Mission (SRTM) and Landsat 8 provided elevation as well as optical and thermal information with almost full coverage of the Earth's surface. However, SRTM and Landsat 8 remote sensing systems have different drawbacks, compared with other

remote sensing devices, which need to be addressed in order to provide acceptable models. One of these problems is the SRTM radar's inability to penetrate tree canopies, which only allows for Digital Surface Models (DSM), instead of Digital Elevation Models (DEM) [10].

The conversion from DSM to DEM requires the detection of forested areas in order to adjust the elevation in those particular pixels. This has a critical impact in research fields like hydrology, where the simulation of water flow is determined according to the land characteristics. Also, forest detection is required in ecological studies like soil classification, forest health tracking, desertification monitoring, among others [16]. The detection of forested areas is not a trivial task and is usually performed with a combination of vegetation detection techniques and human expert intervention, which is tedious and error-prone.

Machine Learning (ML) is a field that offers different techniques to solve complex problems by learning models. During training, classification algorithms adjust models to recognize different patterns in the data to perform classification over the inputs provided. The input data is analyzed automatically during training based on the expected results of each example in the training set. The learned model is then used for the classification task, rather than having to recognize patterns manually or using custom algorithms for each case, reducing the problem of misinterpretation of the data or the omission of unseen patterns. Therefore, ML techniques have a potential application in forest detection.

Different solutions have been proposed based on ML algorithms to solve forest/no-forest classification problems. However, these techniques usually lack accuracy. In this work, we address this problem by means of a Convolutional Neural Network to generate a forest/no-forest mask. The data used for training are based on radar, optical and thermal information provided by SRTM and Landsat-8, using the JAXA Forest/No Forest (JFNF) mask as ground truth.

This paper is structured as follows. Section 2 presents a literature review of techniques in ML and forest recognition. Section 3 describes the data preprocessing. Section 4 exposes the dataset distribution analysis. Section 5 explains the proposed architecture and the classification algorithm. Experiments performed on a case of study are shown in section 6, followed by conclusions and future work (section 7).

2 Literature

Digital Elevation Models are a common representation in Earth Sciences. In particular, hydrological modeling techniques use DEMs to calculate the terrain slope and aspect to predict the divergence or convergence of the water flow [18]. High resolution DEMs can be obtained from sensing devices such as LiDARs mounted on airplanes or unmanned aerial vehicles (UAV), but this involves a high cost for relatively small area coverage. An alternative is the use of cheaper forms of data retrieval with near to global coverage like sensors mounted on satellites or space shuttles, like the SRTM and the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) missions, which provided 30 meter resolu-

tion DSMs. Nevertheless, DSMs are different than DEMs as they include tree canopy in their elevation measures [7]. This affects water flow simulations as forested zones can be interpreted as higher ground areas and thus creating flow divergences, while water actually passes under the tree canopy. Because of this, DSMs need to be converted to DEMs performing a reduction of the measured elevation by an average of tree height on forested pixels.

Different efforts have been made to solve the problems described in SRTM and ASTER DEMs [5, 9]. However, these solutions require manual adjustments or cannot be applied to different zones. The use of multispectral bands data provided by satellite-based sensors like Landsat-8 or Sentinel help with the detection of vegetation through calculating the NDVI as it is exposed by Ganie et al. [6] and Costa et al. [3]. However, NDVI is not accurate enough to discriminate between trees and other forms of vegetation. In order to avoid or reduce human intervention, accurate automatic detection and classification techniques for forest recognition are required.

Machine Learning algorithms are known for their ability to learn patterns from training samples and constructing models for automatic data classification and prediction. Different techniques have been proposed in the area of forest recognition. A Random Forest-based classification system to determine different species of trees in Australia was proposed by Mellor et al. [11]. Pimple et al. [12] present a similar technique to determine the forest types in the north-east mountains of Thailand. A Multilayer Perceptron (MLP) based technique is proposed by Wendi et al. [15] to recognize forested areas from SRTM and Landsat-8 data. Even though these techniques can improve the SRTM DEMs, they were not tested in wide scale areas, they lack accuracy or still require expert intervention. Deep Learning (DL) techniques have been applied to different image processing tasks with better results. In particular, Convolutional Neural Networks (CNN) is a DL technique widely applied in problems like object recognition and classification, noise detection and stereo matching, among many others [2, 13]. This work uses a CNN based system as a binary classifier to create a forest/no-forest (FNF) binary mask, taking SRTM DSM and the different bands of the Landsat-8 satellite as inputs.

3 Data Preprocessing

A dataset was created joining information from the SRTM, Landsat-8 and JFNF projects. Each source of information is accommodated in a cube describing a zone of the Earth. In order to compile a dataset with homogeneous and normalized data, a series of preprocessing steps are required. The data is normalized to the [-1, 1] range, without distorting the difference factor between values, using equation 1.

$$N(\mathbf{v}) = \begin{cases} RNG_m & v < SRC_m \\ RNG_M & v > SRC_M \\ RNG_m + \frac{\left((v - SRC_m) * (RNG_M - RNG_m)\right)}{SRC_M - SRC_m} & otherwise \end{cases} \quad (1)$$

In the equation, $\mathbf{v}$ is the matrix containing the raster values, v is the value to be normalized, RNG_m and RNG_M are the minimum and maximum values for the range, SRC_m and SRC_M are the minimum and maximum valid values for each input type of the data source (e.g. elevation, reflectance, etc).

The selected information from the SRTM and Landsat-8 missions, and the JFNF project come in different presentations, having distinct projection system, resolution, coverage area, etc. In consequence, each information source requires different standard preprocessing steps in terms of raster manipulation, like Digital Number conversion [14], multiple raster merging, reprojection, pixel size redimension, and raster clipping before performing the data normalization mentioned before.

Once each product has been preprocessed, a final cube is created conformed by a first layer with the SRTM normalized data, and the rest of the layers with the different Landsat-8 bands homogenized and normalized. Since the SRTM raster dimension is 3601x3601 and the Landsat-8 raster bands were clipped to this size, the final dataset cube is a 12x3601x3601 dimension structure representing the description of a 1°x1° zone of the Earth.

The FNF binary mask is a 3600x3600 matrix which is used as ground truth to train and compare the results of the proposed architecture. The missing row and column to match the SRTM raster size are simply omitted.

4 Data Analysis

After preprocessing the SRTM, Landsat-8 and JFNF data from 12 different zones covering an area from 25°0'0"S - 53°0'0"W to 28°0'0"S - 49°0'0"W, the training dataset is composed of 89,657,414 No-Forest and 65,862,586 Forest points.

Due to the amount of data available, training with all the points would take a significant amount of time. Therefore, a reduced training set was randomly extracted, whose size was determined by analyzing the distribution of the data. The different layers of the complete dataset were analyzed individually as each one has its own distribution. Smaller samples of different sizes were created randomly and the distribution of each one was compared to the distribution of the full dataset. In order to do that, a histogram of each sample was created, where the number and width of bins (columns in the histogram) were determined with the Freedman & Diaconis [4] rule. The generated histograms resulted in a high concentration of the data points in a relatively short range of values, with a shape similar to a normal distribution.

Even though the width of bins determined was good enough to analyze the data distribution, it also resulted in a large number of bins containing only a few items, as depicted in the left image of figure 1. These bins containing a low number of items represent a problem at the time of comparing a full dataset histogram with a reduced one because they show large relative deviations, which bias the comparison. To overcome this issue, an outlier detection method was used to determine the most representative bins, i.e. those containing a sufficiently large number of items. Following the Iglewicz & Hoaglin [8] work on this topic,

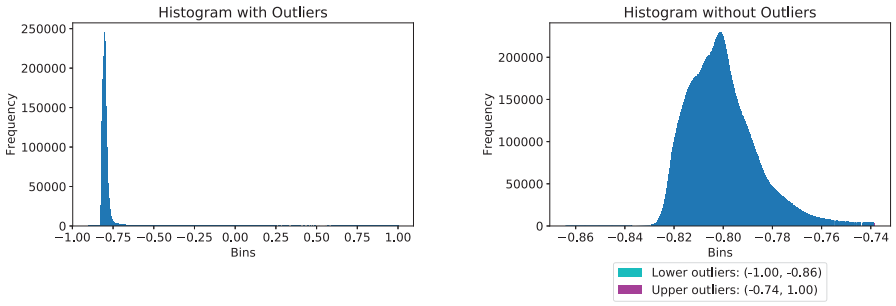

Fig. 1. Forest histograms of the layer 2 using Freedman & Diaconis rule to determine the bins width without outlier removal (*left image*), and the same histogram with outlier removal based on Robust Z-score and Median Absolute Deviation (*right image*).

a robust z-score method with a Median Absolute Deviation is used to determine the importance of each bin in the histogram as shown in equation 2.

$$MAD = median\{|x_i - \tilde{x}|\} \qquad M_i = \frac{0.6745(x_i - \tilde{x})}{MAD} \tag{2}$$

M_i is the score of the histogram bin i, $\tilde{x}$ is the median of the sample and 0.6745 is the $0.75th$ quartile of a standard normal distribution and x_i is the amount of elements in the bin i. Each M_i is then compared with a *threshold* in order to determine the lower and higher values where the bins start being considered as outliers. In our experiments, the suggested *threshold* of 3.5 by Iglewicz & Hoaglin was too restrictive. Instead, we used a *threshold* equal to 4.5 obtaining better results. The resulting histograms after the outlier detection and the determination of the most representative histogram bins range are depicted in the right image of figure 1.

Once the outliers were removed from the histograms, these were used to analyze the reduced samples. The relative deviation of the reduced samples with respect to the full dataset was calculated by comparing the percentage of elements in corresponding bins of the full and reduced datasets, and obtaining the relative absolute difference bin to bin. Once the relative deviations are calculated for each bin, the mean and median error of the histogram for each layer and class was obtained for each reduced dataset. This process was repeated 10 times per reduced sample size that ranged between 5% and 95% obtaining a final average mean and median deviation for each sampling size. Table 1 shows the results for a sample with 30% the size of the full dataset as an example.

The results obtained indicate that a random sample of 30% of the full dataset is the minimum required to ensure a mean and median deviation below 5% and 1% respectively in all the layers of the forest and no-forest classes with respect to the full dataset. The 30% sized dataset has a total of 26,897,225 randomly-selected points for each class, which were then used to train the network.

Table 1. Deviation of 30% sized sample w.r.t. full dataset by class and input layer

Class	Layer type	SRTM	Landsat 8										
		Elev.	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
No Forest	Mean	1.76	4.58	3.82	2.02	1.71	1.48	0.90	0.84	1.75	3.36	2.17	1.74
	Median	0.65	0.59	0.60	0.55	0.49	0.54	0.54	0.50	0.52	0.73	0.68	0.64
Forest	Mean	1.77	2.54	2.39	2.64	2.44	1.07	1.68	1.65	2.28	2.73	1.52	1.70
	Median	0.58	0.54	0.55	0.60	0.57	0.61	0.54	0.53	0.58	0.68	0.65	0.72

5 System Architecture

The model flow and steps are depicted in figure 2. SRTM rasters at a resolution of 30 meters cover a 1°x1° zone, which involves a data matrix of 3601x3601. Additionally, including all the 11 bands of the Landsat-8 satellite clipped at the same size increases the input to a 3601x3601x12 per zone. Because of the size of the input cube, in order to better use the hardware available the input is split in multiple samples of 13x13x12 for each different center pixel.

Since the key piece of information is the pixel at the center of the sample, the rest of the surrounding pixels are considered a 6 pixel neighbor area helping to provide contextual information. In order to be able to process border pixels, each matrix of the zone data cube is padded with a 6 pixels border containing zeros, thus generating a 3613x3613x12 size structure.

In case the model is working in testing mode, all the pixels of the zone input cube are sampled in order and provided in batches to the classification module where the class of each center pixel is predicted. In case the model is set to training mode, only a percentage of the input zone cube is sampled following a random order, but maintaining the same number of samples for each class. The position of each training sample in the zone data cube is also stored in order to retrieve the corresponding ground truth class from the JFNF mask, and then be able to compare the predicted class with the expected class.

With the purpose of providing an optimized classifier, a parameter search was performed over 32 different models, testing alternative architectures with a different number of convolutional layers, feature maps, dense layers, fully con-

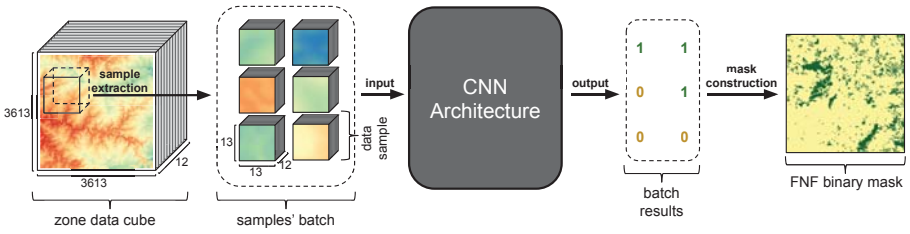

Fig. 2. Proposed architecture input-output model

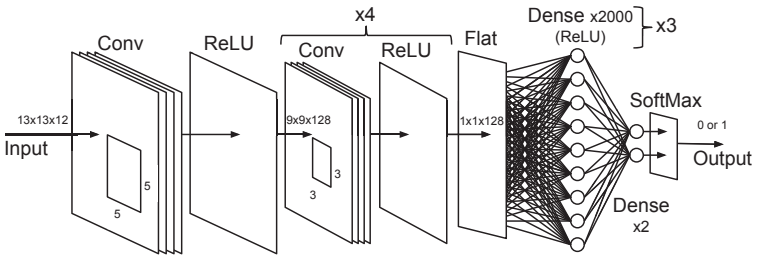

Fig. 3. CNN architecture composed of 5 convolutional layers with 128 feature maps, 3 dense layers with 2000 neurons, and a final SoftMax dense layer with 2 neurons.

nected neurons, kernel sizes at the input layer and alternative optimizers. The architecture described here is the one that presented the best validation accuracy, over a 10 epoch training with a 3 times cross-validation over a 30% reduced dataset.

This architecture is comprised of a series of convolutional layers each one followed by a Rectified Linear Unit (ReLU) activation layer, joining at the end with a fully connected neural network with a SoftMax binary classifier, as depicted in figure 3. Each convolutional layer has a kernel size of 3x3 with no padding and 1x1 stride (except for the first layer which has a 5x5 kernel), generating 128 feature maps. The model predicts the class of a certain pixel by evaluating the twelve different input channels corresponding to the SRTM DSM and Landsat-8 bands along with the neighbor pixels. According to the number of convolutional layers, their kernel size, padding, and size, the minimum input size of 13x13 window is provided creating a final input of 13x13x12. After the five Convolutional/ReLU layers a flattening layer is added in order to generate a vector of features to be fed into a fully connected neural network, which in turn is composed of 4 layers: three layers with 2000 neurons and ReLU activation function, and a last layer with 2 neurons using a SoftMax activation function. The model provides a single value output indicating if the pixel in the center of the input window corresponds to the Forest or No-Forest class.

The classifier is trained using a binary cross-entropy loss function and Stochastic Gradient Descent optimization technique.

The construction module is executed only in testing mode once the classifier module has finished predicting all the pixels' class. It simply reshapes the single vector of results to a matrix of size 3601 x 3601 creating in this way a mask where each position corresponds with a position of the SRTM raster used in the input.

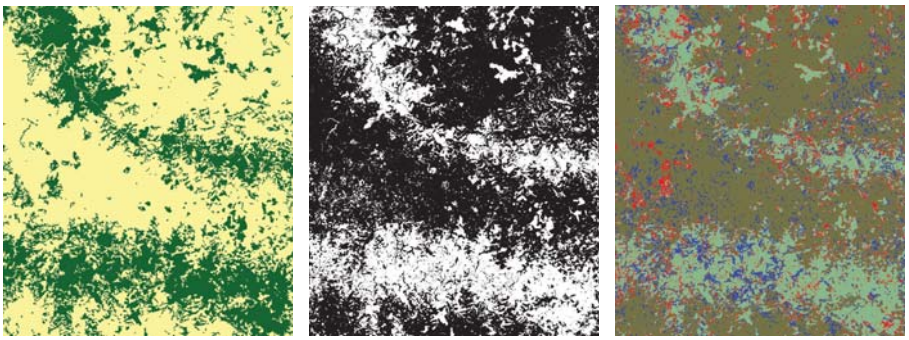

Fig. 4. Zone 13 of the JFNF mask (*left*), our FNF mask (*center*) and comparison (*right*) where correctly predicted points are shown in brown and green (No-Forest and Forest respectively), while errors are shown in blue and red respectively.

6 Case Study

The experiments were performed on an AMD Ryzen 1700 CPU with 32 GB DDR–4 2400 MHz RAM and a NVidia Titan Xp. The source code and the data used in this work are publicly available[4].

The training samples were created by obtaining the 13x13x12 data cubes at each position of the dataset and were provided in batches of 512 samples to the training algorithm. A k-fold cross-validation training schema was used with $k = 5$. A selection of 45% of the dataset points was randomly selected maintaining the balance between No-Forest and Forest points. Since KFold splits 45% of the points into 5 parts, 36% of the points are selected for training/validation and 9% for test on each of the 5 runs. Then, the training/validation set is split in 85% for training and 15% for validation, generating a 30.6%/5.4% selection from the full dataset respectively and thus, maintaining a higher than 30% dataset selection for training as it was analyzed in the section 4. The network was trained over 30 epochs with a learning rate of 0.01 and default parameters following the recommendations for Stochastic Gradient Descent in [1]. In our tests, other optimizers like Adam produced vanishing gradients so they were discarded.

The classification module obtained an average test accuracy of 91% during training. The results of the KFold cross-validation in terms of average precision, recall and F1-score (with their standard deviations) for the No-Forest class were 91.4% (1.5%), 91.4% (1.5%) and 91.2%(0.4%), while the Forest class obtained 91.4% (0.9%), 91.4% (1.5%) and 91.4% (0.5%) respectively. Then the model was tested against the test zone 13 composed of 8,546,237 No-Forest and 4,413,763 Forest points with an accuracy of 87.36% taking 352 seconds.

The JFNF mask and the one produced by our network for zone 13 are shown and compared in the figure 4. One thing to notice is that in some cases the

[4] https://github.com/labsin-uncuyo/py-cnn-geo

Fig. 5. Small area of the comparison between JFNF mask and our FNF mask showing multiple errors in our results in blue and red (*left image*), but correctly detecting real forested areas omitted by the JFNF mask (*right image*).

network was capable of recognizing real forest points that were not correctly labeled in the JFNF mask (figure 5).

7 Conclusion and Future Work

In this paper, we presented a CNN architecture capable of learning to recognize forested areas from different inputs obtained by remote sensing devices. The model generated a binary mask with an 87.36% accuracy compared with the JAXA FNF mask in less than 6 minutes for 1°x1° coverage areas. After a detailed analysis of the results, we found that the network is able to recognize forested areas omitted in the ground truth mask, denoting a potential benefit of using our approach even for improving the available data. Also, these misclassified points in the ground truth mask could be introducing noise in the training phase of the network, so we think the results can be further improved by using better data for training.

Future work includes testing additional network models, training and optimization parameters in order to find better classification architectures. Also, we will explore different post-processing techniques for improving the quality of the binary mask and reducing the noise. In addition to this, the proposed solution should be tested in areas with an imbalanced proportion of Forest and No Forest points, or different land characteristics in order to assess the strengths and weaknesses of the model in more difficult scenarios. An analysis of the significance of input layers on the result can help to determine unneeded input data and result in faster and more accurate execution of the network. Finally, other satellite-based data inputs can be explored, like Sentinel mission data, to verify the model's sensitivity to different data sources.

Acknowledgments. We want to gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

1. Bengio, Y.: Practical Recommendations for Gradient-Based Training of Deep Architectures. In: Montavon, G., Orr, G., Müller, K. (eds.) Neural Networks: Tricks

of the Trade. Lecture Notes in Computer Science, vol 7700, pp. 437–478. Springer, Berlin, Heidelberg (2012)
2. Caffaratti, G., Marchetta, M., Forradellas, R.: Stereo matching through squeeze deep neural networks. Inteligencia Artificial 22(63), 16–38 (Feb 2019)
3. Costa, S., Santos, V., Melo, D., Santos, P.: Evaluation of landsat 8 and sentinel-2a data on the correlation between geological mapping and ndvi. In: 2017 First IEEE International Symposium of Geoscience and Remote Sensing (GRSS-CHILE). pp. 1–4 (June 2017)
4. Freedman, D., Diaconis, P.: On the histogram as a density estimator:L2 theory. Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete 57(4), 453–476 (1981)
5. Gallant, J., Read, A., I. Dowling, T.: Removal of tree offsets from srtm and other digital surface models. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XXXIX-B4, 275–280 (07 2012)
6. Ganie, M., Nusrath, A.: Determining the vegetation indices (ndvi) from landsat 8 satellite data. International Journal of Advanced Research 4, 1459–1463 (09 2016)
7. Gesch, D., Oimoen, M., Danielson, J., Meyer, D.: Validation of the aster global digital elevation model version 3 over the conterminous united states. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLI-B4, 143–148 (06 2016)
8. Iglewicz, B., Hoaglin, D.C.: How to Detect and Handle Outliers (1993)
9. kuan Liu, J., Liu, D., Alsdorf, D.: Extracting ground-level dem from srtm dem in forest environments based on mathematical morphology. IEEE Transactions on Geoscience and Remote Sensing 52(10), 6333–6340 (Oct 2014)
10. M. Lillesand, T., Kiefer, R., W Chipman, J.: Remote Sensing and Image Interpretation (Seventh Edition) (01 2015)
11. Mellor, A., Haywood, A., Jones, S.D., Wilkes, P.: Forest classification using random forest with multisource remote sensing and ancillary gis data (2014)
12. Pimple, U., Sitthi, A., Simonetti, D., Pungkul, S., Leadprathom, K., Chidthaisong, A.: Topographic correction of landsat tm-5 and landsat oli-8 imagery to improve the performance of forest classification in the mountainous terrain of northeast thailand. Sustainability 9 (02 2017)
13. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., C. Berg, A., Li, F.F.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision 115 (09 2014)
14. USGS: Landsat 8 (L8) Data Users Handbook (LSDS-1574), https://www.usgs.gov/media/files/landsat-8-data-users-handbook
15. Wendi, D., Liong, S.Y., Sun, Y., Dung Doan, C.: An innovative approach to improve srtm dem using multispectral imagery and artificial neural network. Journal of Advances in Modeling Earth Systems 8 (04 2016)
16. Wilson, J.: Digital terrain modeling. Geomorphology 137 (01 2012)
17. Wilson, J., Gallant, J., Hutchinson, M.: Future directions for terrain analysis, pp. 523–527 (01 2000)
18. Zhou, Q., Pilesjö, P., Chen, Y.: Estimating surface flow paths on a digital elevation model using a triangular facet network. Water Resources Research 47(7) (2011)