

Neural Network Plugin for Definiens Developer 7.0

Short Reference

F. Bachmann

Geomonitoring Group, Institute for Mine-Surveying and Geodesy,
Technische Universität Bergakademie Freiberg

Neural Networks have been well established in pixel-based analysis of remote sensing image data. Due to latest progress regarding object-based image analysis, which mostly uses fuzzy sets for classification purposes, the goal of automation is attained respectively reducing time and knowledge effort, hence techniques are required facilitating this.

This plugin brings *Feed Forward Networks* and *Class Based Networks* for testing its object based capabilities into Definiens. It has been developed with the Definiens Developer SDK¹ and IT++ library², which can itself make use of Intel's Math Kernel Library³.

The Geomonitoring Group of Technische Universität Bergakademie Freiberg holds all rights concerning the software. The software is free to distribute.

Contents

1	Installation	1	2.1.1	Feed Forward Network	3
2	Usage	1	2.1.2	Feed Forward against other	3
2.1	Neural Networks	2	2.1.3	Class Based Network	3

1 Installation

Copy the `DIAProcAlgrNeuralNet.dll` to Definiens Plugin Directory, by default this is located at `'C:\Program Files\Definiens Developer 7.0\bin\plugins'`.

Run Definiens and append a new algorithm in the **Process Tree**, the plugin algorithms are initially hidden under the available algorithms, these can be found selecting **'more'** (Fig. 1a), they will appear as new item **'Neural Networks'** (Fig. 1b).

2 Usage

Both the paradigm of neural networks (Bishop, 1995) and its application to supervised classification (Canty, 1999) are well known.

The preparation to run a supervised training with neural network implies the setup of training data, this can be done by selecting class samples with the sample editor (Fig. 2b) in a segmented

¹ <http://www.definiens.com/>

² <http://itpp.sourceforge.net/>

³ <http://www.intel.com/cd/software/products/asm-na/eng/307757.htm>

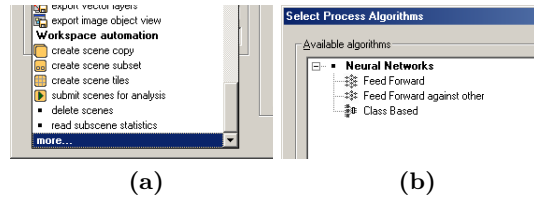


Fig. 1: Process tree in Definiens Developer:
Process Tree > Append New > More > Neural Networks

scene (Fig. 2a). A chessboard segmentation with object size one is similar to the pixel-based case.

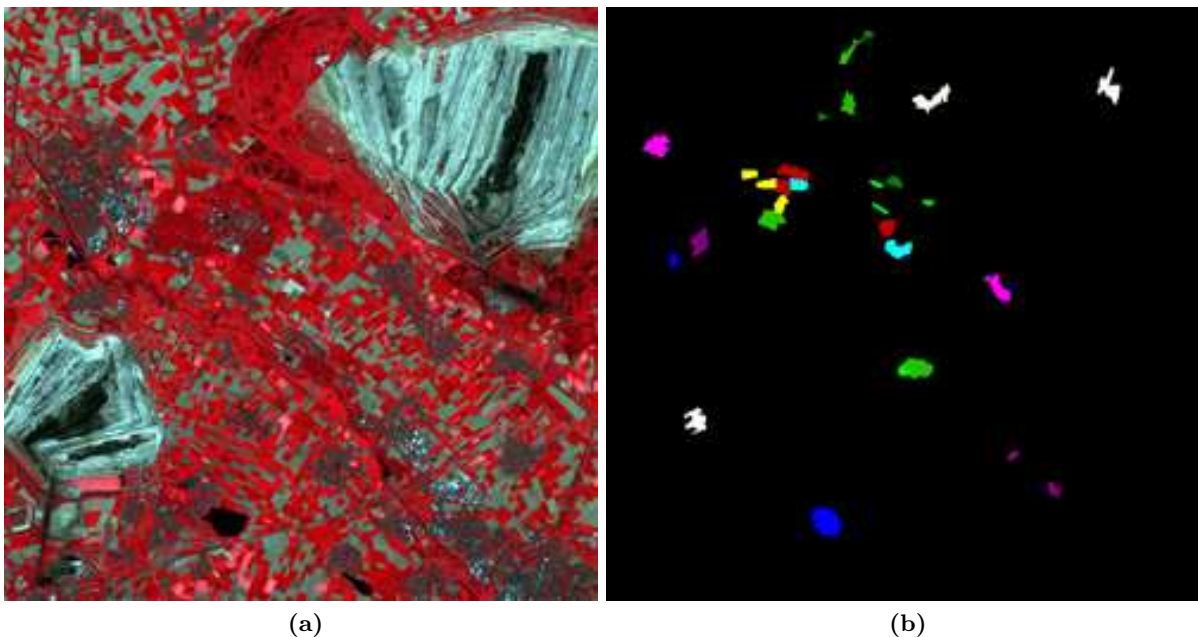


Fig. 2: Sample Scene: (a) ASTER Image Scene over Juelich, Germany, on May 1st, 2007. (b) Selected Samples of 10 Classes for Network Training: cereal (red), industry (violet), pasture (light green), deciduous forest (green), coniferous forest (dark green), rapeseed (yellow), beet (cyan), residential (pink), open-pit mining (white), water (blue).

2.1 Neural Networks

The plugin contains three learning algorithms (Haykin, 1998; Canty, 1999) and two mixed setups for a two-layered feed forward network (FFN):

- Backpropagation
- Kalman Filter Training
- Scaled Gradient Conjugated (SCG)
- Kalman Filter + Backpropagation
- Kalman Filter + SCG

where the last both are intended to find the global minimum of the Error Function. The randomly initiated weights and learning result in a non-unique classification, but it can be shown that the accuracy of a network can be approximated in any order (Bishop, 1995).

2.1.1 Feed Forward Network

The algorithm setup (Fig. 3) requires a set of training data $\{(g(v), l(v)) | v = 1 \dots n\}$ where $g(v)$ is a biased vector of observations represented through class features ('Select Features') and $l(v)$ its associated labeled class out of a set of $1 \dots n$ classes using the marked samples (see Fig. 2b) ('Use class samples'). The dimension of input signals $g(v)$ and output signals $m(v)$ is given, the dimension of the '1st hidden layer' needs to be determined, as well as the number of training iterations measured in 'epochs', where one epoch is the number of training samples. 'Softmax' activation is the default output signal, setting up it to false will change it to normal sigmoidal activation.

Additionally, the Image Object Domain in where the classification should be executed can be defined and the training is interruptible.

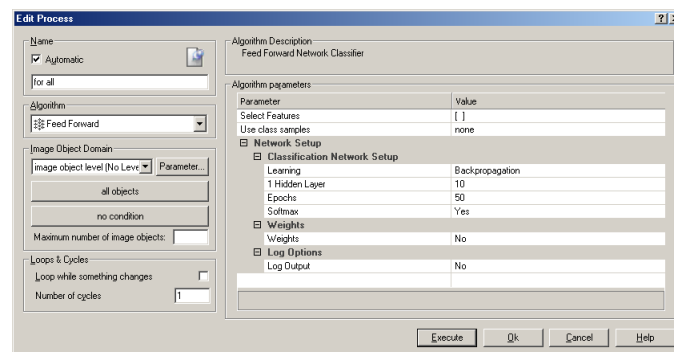


Fig. 3: Feed Forward Network Setup in Definiens Developer

The Training of the Network could be saved and loaded in a Weights-file.

Configuration of logging options enables reporting during the network training to a specified file. 'Forward Pass' will print the training vectors $g(v)$ and $l(v)$, where 'McNemar' splits the training data into a set of two, one for the training, and the other is forwardpassed after training and compared to its training labels. 'Epochs' prints the cost history of learning in the given intervall.

2.1.2 Feed Forward against other

The plugin provides another training mode (Fig. 4), where one certain 'Class' is trained together with selected classes ('Use class samples') (Fig. 5). The activation signal of 'is Class' state is stored into an Object Variable ('Class Variable') and additionally, if sigmoidal activation is choosen, the 'is not Class' state ('not Class Variable'), since Softmax outputs $\sum m(v) = 1$, thus $m(v)_{isnot} = 1 - m(v)_{is}$. The configuration of the network is done like in 2.1.1.

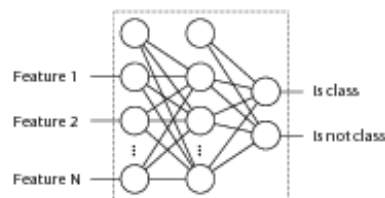


Fig. 4: Network topology of Feed Forward against other

2.1.3 Class Based Network

The class based approach (Marpu et al., 2008a) is based on the idea of class separability (Marpu et al., 2008b). A class is described by a set of features. As these features are often difficult to figure out, tools for automated feature extraction are needed. We have implemented a procedure

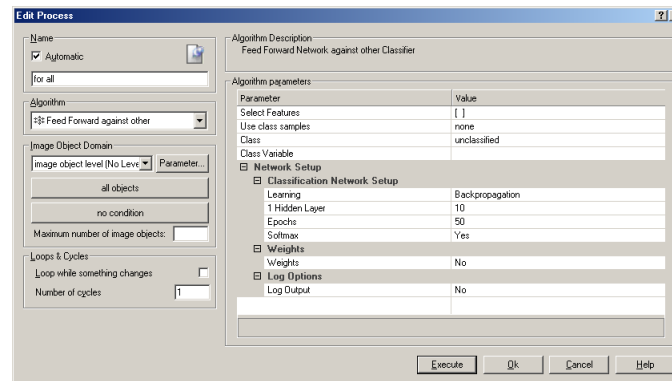


Fig. 5: Feed Forward against other Setup in Definiens Developer

to automatically define a model for fuzzy classification. A class-based Neural Network is built up by coupling FFNs that identify each class based on preselected features. (Fig. 6).

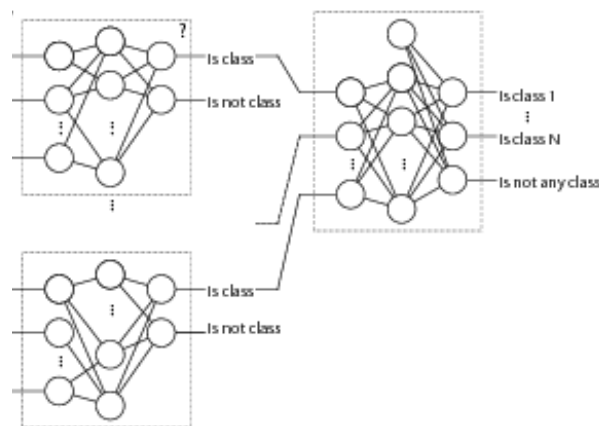


Fig. 6: class based network topology

The configuration of the algorithm (Fig. 7) is similar to 2.1.1, each Class Based FFN has the same architecture and learning configuration. The input has to be defined for each class, the training results of each network-node is routed into the classification network.

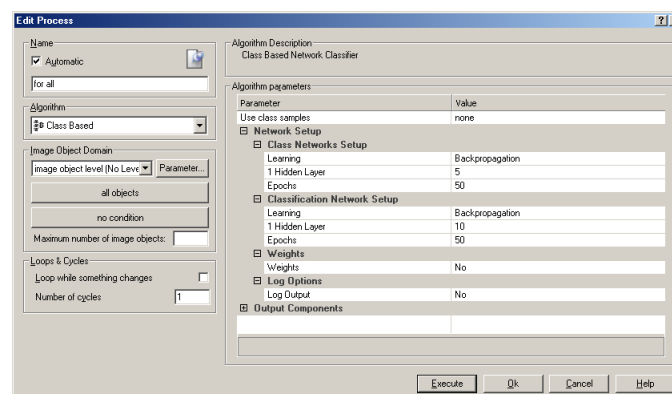


Fig. 7: CBN Setup in Definiens Developer

References

C. M. Bishop, *Neural Networks for Pattern Recognition*. New York, NY, USA: Oxford University Press, Inc., 1995.

-
- M. J. Canty, *Fernerkundung mit neuronalen Netzen*. Renningen-Malmsheim: Expert Verlag, 1999.
- S. Haykin, *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.
- P. Marpu, F. Bachmann, and I. Niemeyer, "A class dependent neural network architecture for object-based classification," in *GEOBIA 2008*, Calgary, 6-7 August 2008.
- P. Marpu, I. Niemeyer, S. Nussbaum, and R. Gloaguen, *Object-Based Image Analysis Spatial Concepts for Knowledge-Driven Remote Sensing Applications*, ser. Lecture Notes in Geoinformation and Cartography. Berlin: Springer, 2008, ch. A Procedure for automatic object-based classification, pp. 169–185.