

The Backpropagation software (v1.0)  
Mattias Wahde  
[mattias.wahde@me.chalmers.se](mailto:mattias.wahde@me.chalmers.se)  
Chalmers University of Technology

The Backpropagation (v1.0) software, which was written in Delphi 5, implements the backpropagation training algorithm (using stochastic gradient descent) and applies it to two-layer networks of arbitrary size. The program is started by clicking on the Backpropagation icon. The following window appears:

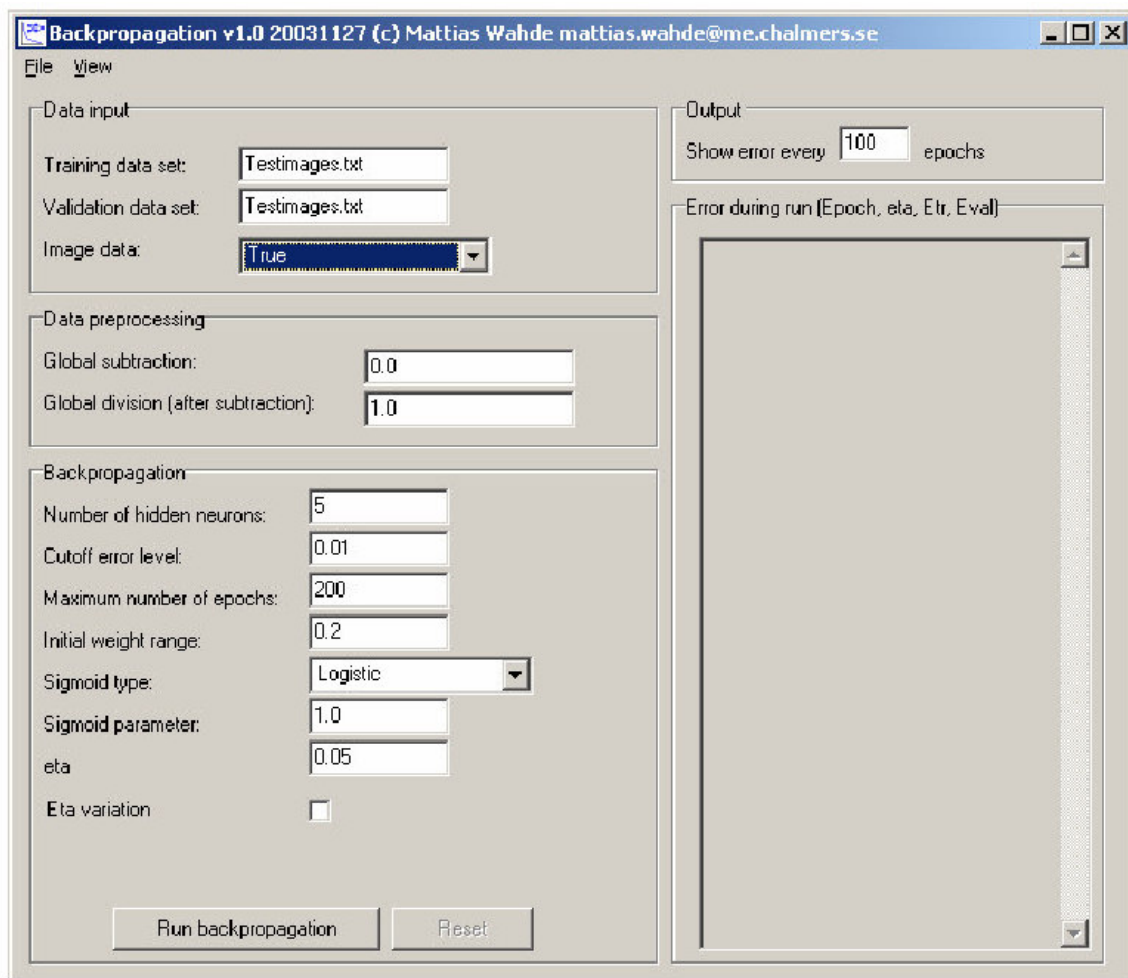


Fig. 1. The main window of Backpropagation , v1.0.

Backpropagation (v1.0) can read data either in the standard format (see below) or as a matrix corresponding to an image. If *Image data* is set to *False*, the program reads data in the standard format. However, if *ImageData* is set to *True*, the program expects image data.

### Standard data format

The data sets should be text files of the following form: on the first line, the name of the data set should be given. The second line should contain the number ( $N$ ) of input signals, and the number of output signals. Then, the data set should follow. On each line, the  $N$  input signals should be given first, and should be followed by the desired output signals.

Example:

```
USD_SEK_training_data
5 1
7.4485 7.4319 7.4144 7.4212 7.4117 7.3856
7.4319 7.4144 7.4212 7.4117 7.3856 7.4024
7.4144 7.4212 7.4117 7.3856 7.4024 7.4226
7.4212 7.4117 7.3856 7.4024 7.4226 7.4813
7.4117 7.3856 7.4024 7.4226 7.4813 7.5039
7.3856 7.4024 7.4226 7.4813 7.5039 7.5170
7.4024 7.4226 7.4813 7.5039 7.5170 7.5309
7.4226 7.4813 7.5039 7.5170 7.5309 7.5529
etc.
```

**Table 1. An example of a data set in the standard format.**

### Image data

An example of the format is shown in Table 2 below. The data set shown in the table contains two images with  $9 = 3 \times 3$  pixels. The first row contains the name of the data set, and the number of rows and columns in each image are given on the second row. The third number on the second row indicates the number of output signals (1, in this case). The desired output corresponding to each image is given on the row immediately below the image. In the example in Table 2, the desired outputs are 0.0 and 1.0, respectively. Note that image data must be provided as gray scale values in the range  $[0.0, 1.0]$ , where 0.0 corresponds to black and 1.0 to white. 0.5 yields an intermediate gray value.

```
Testimages
3 3 1
0.0 0.0 0.0
0.0 1.0 1.0
0.0 0.0 1.0
0.0
1.0 1.0 1.0
1.0 0.0 1.0
0.0 1.0 1.0
1.0
```

**Table 2. An example of an image data set, containing two images.**

## Examples

### Example 1: A Boolean function

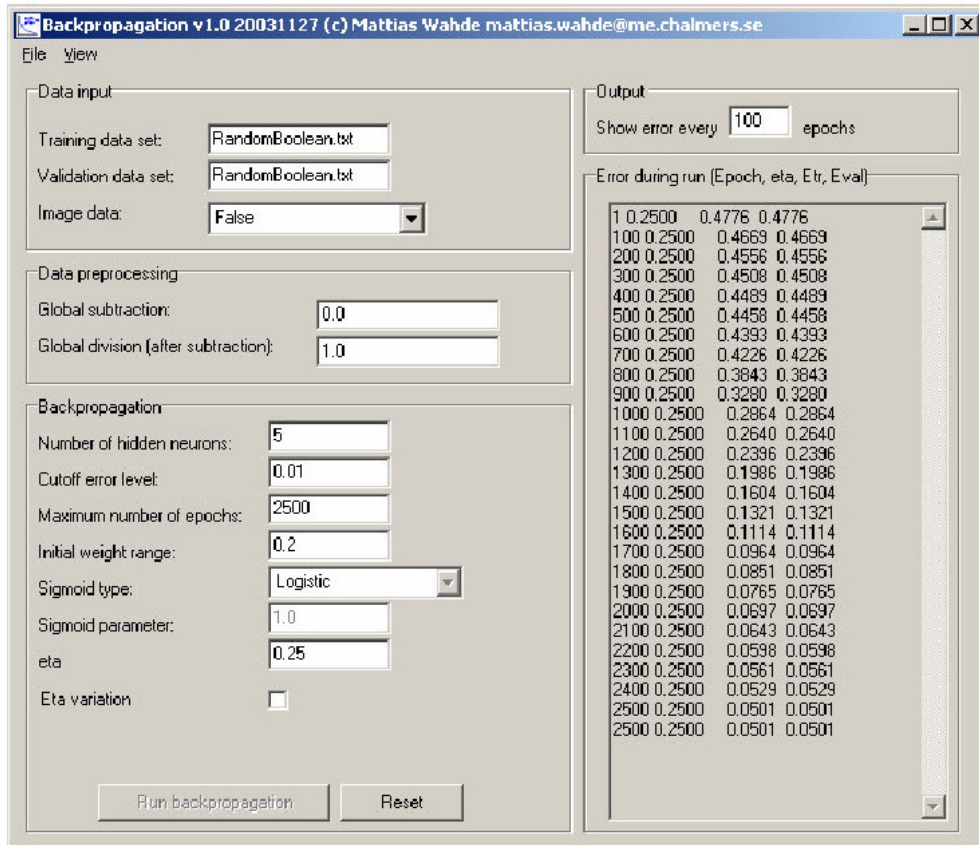
As can be seen from Fig. 1, the user may choose a training data set and a validation data set, by providing the file names. It is also possible to preprocess the data. This is often needed, since the sigmoid function generally limits the range of the output (i.e. to  $[0,1]$  or  $[-1,1]$ ). The preprocessing consists of two steps: first a constant (global subtraction) is subtracted from all elements in the data set; Next, all elements in the data set are divided by another constant (global division). By default, no preprocessing is performed. For the backpropagation algorithm, several parameters must be set, namely

- (1) The number of neurons in the middle layer (the number of input elements and the number of neurons in the output layer are, of course, specified by the data set).
- (2) The cutoff error level, at which point the training is stopped.
- (3) A maximum number of training epochs, used for stopping the training in cases where the cutoff error level is not reached.
- (4) The sigmoid type: linear, logistic, or hyperbolic tangent. The logistic sigmoid limits the range of the output to  $[0,1]$ , whereas the hyperbolic tangent sigmoid limits the range to  $[-1,1]$ .

*Note: the hyperbolic tangent sigmoid is not implemented in this version of the program. Instead, data given in the range  $[-1,1]$  should be rescaled to  $[0,1]$  by subtracting  $-1$  and dividing by  $2$ , i.e. global subtraction =  $-1$ , global division =  $2$ .*

- (5) The sigmoid parameter ( $c$ ), which determines the slope of the sigmoid around  $z=0$ .
- (6) The learning rate parameter ( $\eta$ ), which is constant in this program

Finally, the frequency of error (training and validation) output must be specified. In order to start the training procedure, press the **Run backpropagation** button. The errors are shown in the box on the right side of the window. There is no stop button the learning proceeds until one of the two termination criteria above is reached. An example is shown in Fig. 2.



**Fig. 2. Results from a run on the RandomBoolean data set.**

Here, a 4-4-1 network was trained to represent the Boolean function given by the following truth table

---

Random_Boolean_function				
4	1			
0.00	0.00	0.00	0.00	1.00
0.00	0.00	0.00	1.00	0.00
0.00	0.00	1.00	0.00	0.00
0.00	1.00	0.00	0.00	1.00
1.00	0.00	0.00	0.00	1.00
0.00	0.00	1.00	1.00	1.00
0.00	1.00	0.00	1.00	1.00
1.00	0.00	0.00	1.00	0.00
0.00	1.00	1.00	0.00	1.00
1.00	0.00	1.00	0.00	1.00
1.00	1.00	0.00	0.00	0.00
0.00	1.00	1.00	1.00	1.00
1.00	0.00	1.00	1.00	1.00
1.00	1.00	1.00	0.00	0.00
1.00	1.00	1.00	1.00	1.00

---

**Table 3. A Boolean data set.**

This is a data set provided in the standard format. Thus, *image data* was set to *False*.

## Example 2: Image data

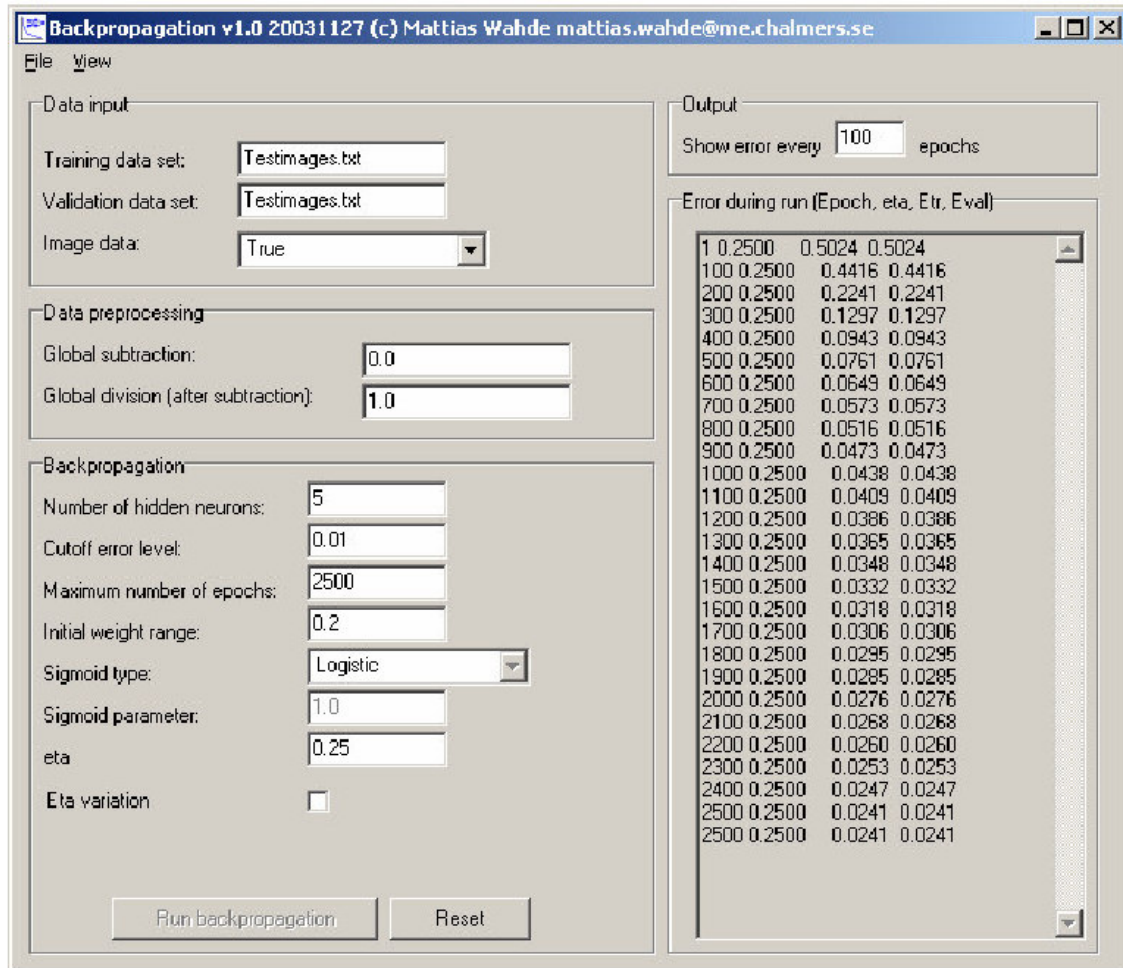
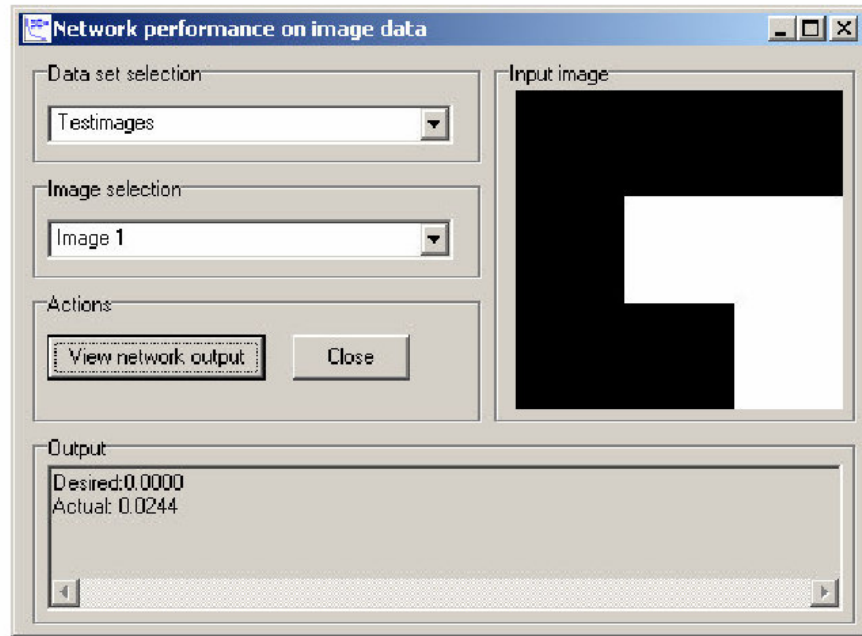


Fig. 3. Results from a run on the TestImages data set.

In this case, the data set TestImages.txt was used, and the *Image data* property was thus set to *True*. When the training has been completed, the output can be examined by selecting **Final network performance** under the **View** menu. However, in the case of image data, another option exists as well. If **Final network performance on image data** is chosen, the following window appears:



**Fig. 4.** The desired and actual network outputs, as well as an image of a kanji sign used in the training of the network.

Here, an image can be selected from either the training data set or the validation data set. If the **View network output** button is clicked, the desired and actual network outputs, as well as the image, are shown in the window as illustrated in Fig. 4. (*Note: do NOT use this window UNLESS the data is of the Image type*).