Neural Network Development Tool

Evaluation version 1.2 Björn Saxén 1995

General

- pattern file
- setup file
- output file
- log file
- test file
- general settings
- command line arguments
- graphical presentation of outputs
- network state
- log table
- graphical presentation of training progress

MLP networks

- MLP network configuration setup
- MLP training setup

Help on the actual topic is achieved by pressing F1 at any time in NNDT

The file MANUAL.ZIP contains a user's guide for NNDT in PostScript format.

NNDT - Neural Network Development Tool

The NNDT software is as a tool for neural network training. The user interface is developed with MS Visual Basic 3.0 professional edition. DLL routines (written in C) are used for most of the mathematics. The program can be run on a personal computer with MS Windows, version 3.1.

Evaluation version

This evaluation version of NNDT may be used free of charge for personal and educational use. The software certainly contains limitations and bugs, but is still a working version which has been developed for over one year. Comments, bug reports and suggestions for improvements can be sent by e-mail to:

bjorn.saxen@abo.fi

Remember, this program comes free but with no guarantee!

Features and methods

NNDT includes a routine for graphical presentation of the output signals, residuals, weights and node activations during run. Network weights and node activations are easily examined in tables and the weights can be modified by the user.

The network algorithms implemented are of the so called supervised type. So far, only algorithms for MLP networks of feed-forward and recurrent type are included. The training requires a set of input signals and corresponding output signals, stored in a file referred to as pattern file. This is the only file the user must provide.

Optionally, parameters defining the pattern file columns, network size and network configuration may be stored in a file referred to as setup file. When saved after a network training, the setup file also includes the achieved network weights. The use of setup files makes the interactive work faster.

A test file can be used to evaluate the generalisation performance during training.

The user can select the option output file to get the result from each time step saved for later use.

NNDT is developed mainly for interactive use. However, the program can be run in a non-interactive "batch" mode when started with appropriate command line arguments. Several instances of NNDT can be run simultaneously.

Pattern file

The pattern file supplied by the user holds input signals and corresponding output signals for the supervised learning. All input and output signals for one pattern should be written on the same line in the file. Each line should end with a carriage return. The pattern file is chosen in the pattern file setup window (accessed from the setup menu) where also the locations of the columns for input and output signals are specified as well as the number of header lines to be skipped in the beginning of the pattern file. All information about the pattern file is stored in the setup file.

The numbers (columns) on one line in the pattern file may be separated by spaces and/or tabs. A pattern file can e.g. be created with MS Excel. In Excel, save the worksheet containing the patterns as a text file.

It should be noted that data pre-treatment, specified in the MLP setup window, may result in training patterns different from those in the pattern file.

Demo files

Two pattern files for demonstration are delivered with the NNDT installation files. The file DEMO1.DAT consists of 64 patterns of x, sin(x), cos(x) and tan(x), where x goes from 0 to approx. 2π. Already a feed-forward network of moderate size should be able to describe sin(x) and cos(x) as functions of x, whereas tan(x) may be a more complicated task. A recurrent network without inputs (DEMO1.MLP) is capable of finding the circle described by sin(x) and cos(x).

The file DEMO2.DAT consists of 600 patterns of x(t), x(t-6), x(t-12), x(t-18) and x(t+85) where **x** is a chaotic time series given by a Mackey-Glass differential equation. The prediction of x(t+85) may require quite large network sizes.

Setup files for the demo data files are given (DEMO1.MLP, DEMO1B.MLP, DEMO2.MLP). Start by loading the demo setup file and then make the configuration changes you wish. DEMO1.MLP is loaded automatically the first time you run NNDT after installation.

Setup file

The setup file is optional, and it is chosen from the file menu. A setup file makes work faster when similar runs are carried out subsequently. The file holds all information required about network size and configuration. When created based on a trained network, the file also contains the network weights.

The setup file also holds information about the pattern file name and the location of input and output columns in this file.

When NNDT is started, the latest setup file selected (in previous run) is automatically read.

The setup files DEMO1.MLP, DEMO1B.MLP and DEMO2.MLP delivered with NNDT are used for the demo examples.

Output file

The output file is optional, and it is normally used when further analysis or graphical presentation of the data is needed. The output file is created by feeding the training or test patterns to the network immediately after training (or evaluation), so the output file must be selected before training is started.

The output file name and the variables to be written to the file are specified in the output file setup window, accessed from the setup menu.

Log file

The log file is optional and holds the same information as the log table, but the log file is especially suited for non-interactive runs.

The log file name and the variables to be written to the file are chosen in the log file setup window, accessed from the setup menu. A new line is appended to the log file after each iteration, thus, the log file must be selected before training is started.

Test file

The test file is optional, and it is chosen in the test file setup window, accessed from the setup menu. The test file option enables evaluation of the generalisation properties of the network throughout training by means of observations which are not included in the training data. The test data is fed through the network after each iteration in the training and the rms error for the test patterns are shown. The test error is saved in the log table and a graphical presentation is given in the training progress window. The user can examine node activations for test data in the network state window and in the graph window. The test result can also be saved in the output file .

The observations in the test file must written in analogy to the pattern file, i.e. the columns for the variables should be same as in the pattern file. The test file observations are treated similarly to the training data, i.e., same parameters for scaling of pattern file values and data pre-filtering are used. However, a separate number of header lines can be specified for the test file, and, the entire test file is used even if the number of training patterns is limited.

General settings

The following switches appear in the main window. Some of them may only be available for certain network types.

Single step mode

Checking this box forces the training to be paused at frequent intervals.

Train net

If this box is not checked, no training occurs and the network is only evaluated with the data in the pattern file.

Network type (setup menu)

The network type is selected in the setup menu

Multitasking (options menu)

Large networks may require considerable training times. The grade of multitasking, i.e. how often NNDT checks if other events are in the queue, is set by the multitasking switch. Normally, full multitasking should be used. Limiting the grade of multitasking makes the training a little faster but may slow down the system quite a bit.

Command line arguments

NNDT can be started with optional command line arguments which enable non-interactive runs. The syntax for the command line is:

nndt [setup_file] [/st] [/sa] [/ex]

All parameters in brackets are optional and may be given in any order.

- setup_file is the name (including path if it is not the default) of the setup file
- /st specifies that the program (training) is started immediately (so the user does not have to press the "start" button)
- /sa specifies that the setup (including weights) is saved to the setup file automatically after finished training
- /ex specifies that the program is exited after finished training (and possible automatical setup saving)

When an output file and/or a log file (specified in the setup file) are used, a non-interactive run can be analysed later.

Several instances of NNDT can run simultaneously.

Pattern file setup window

In this window, the user can specify the name of the pattern file, the number of input and output signals and the location of the columns in the file, as well as the number of header lines to be skipped in the beginning of the file. (At least one output target signal is required, but recurrent networks do not necessarily need input signals.)

Scale factors can be specified for inputs and outputs. Input signals are scaled before they are fed to the network and the desired outputs signals are scaled before compared to network outputs.

Always reread file

If this box is checked, the pattern file (and the test file, if any) is always read before network training or evaluation. This options is needed if the user makes changes in the pattern file (or test file) while NNDT is loaded.

Output file setup window

In the output file setup window, the user can specify if an output file should to be used, select the file name, choose if train or test data shall be saved, and select the variables to be written to the output file.

Variables that can be written to the output file are input signals, desired output signals, network output signals and internal node activations.

Log file setup window

In the log file setup window, the user can specify if an log file should to be used, select the file name and select the variables to be written to the output file.

The variables to be written to the file are specified by the user, possible choices are
- real time (from computer´s clock)
- iteration number
- SSQ error
- rms error
- rms error for test set (if any)
- Marquardt parameter

Test file setup window

In this window, the user specifies if a test file shall be used and the name of the file. The number of header lines to be ignored in the beginning of the file is also specified. Scale factors specified for the pattern file (in pattern file setup window) and parameters for pre-treatment of training data (given in the MLP setup window) are used also for the test file. However, the entire test file (except the header lines) is always used, even if the number of training patterns is limited.

A single file may be used both as pattern file and test file. By specifying a maximum number of patterns to be formed before training (in the MLP setup window) only the first part of the file is used for the training patterns. The rest of the file is used for test patterns by specifying an appropriate number of header lines for the test file.

Graphical presentation

In the performance graph window, accessed from the show menu, the network activations, weights etc. are illustrated graphically.

This window can be kept open during network training.

Options menu

Copy graph - The graph can be copied to Windows clipboard as a metafile.

Plot menu

Variables - Choose between:

* Outputs (network and desired) against pattern index
* Residuals against pattern index
* Weights
* Special: one or several node activations can be plotted against any node activation or against pattern index. See plot specification.

Plot density - Setting a higher number gives faster but less accurate plots.

Train data / Test data - Choose if train or test data shall be used for the plot (does not affect plot of weights)

Redraw interval

The redraw interval specifies the number of iterations between graph updating. The same update interval is used for the training progress plot

Plot specification

In the window for specification of plot variables, accessed from the plot/variables menu in the graph window, the user assigns variables for both axis on the plot. One or several node activations can be plotted against any node activation or against pattern index.

A variable is selected by simply dragging from the picture of network nodes and dropping on the desired axis. The x-axis can hold only one variable, whereas the y-axis can hold several. A variable on the y-axis is replaced when a new variable is dropped on it. To add y-axis variables, drop them on the "free region" of the axis. Variables are removed from the y-axis by dragging them to refuse bin.

Network state window

The network state window, which is accessed from the show menu, shows internal states (node activations) and values of the weights in the network. Weights (and initial states) can be initialised by a random number generator within the ranges specified in the parameter initialisation window.

This window can be kept open during network training.

For MLP networks, the network state window is used for:

Examination and modification of weights

Network weights and initial values for the fictitious input nodes can be viewed at any time, also during iteration. The values can also be changed, but not during iteration. Although the values shown in the table are rounded to three significant digits, full precision is used when editing the values. The clipboard in Windows enables import to, or export from, other applications.

The first lines in the weight table show the weights for the connections to the first hidden layer (if any) and below are shown the weights to any upper hidden layers and to the output layer. If by-pass (input-output) connections are used, the weights for these are shown in the last lines in the table. Each line contains the weights for all connections leading to one node. The first element (column) is the bias, except for in the lines showing by-pass connections where the the first element is the weight to the first input node.

Examination of node activations

Node activations can be viewed during and after training. The pattern number for which the activations are to be shown is given by the user. The activations are updated after each iteration in the training. In the options menu, the user can choose between train data and test data.

Parameter initialisation window

A network training procedure is normally started from randomly assigned values for network weights and bias terms (and initial states for recurrent networks). This assignment is made in the parameter initialisation window accessed from the options menu in the network state window.

The user can initialise the weights leading to each layer (as well as the initial states) separately and give separate initialisation ranges. The seed for the random number generator is also given by the user; subsequent initialisations give identical weights if the seed is kept constant.

The new parameters are immediately shown in the network state window.

Log table window

After each iteration in training, the time (from the computer's clock), iteration number, SSQ, rms error and the Marquardt (lambda) parameter are saved in a list. If a test file is used, the rms error for the test set is also written to the list. The list can be analysed in the log table window, accessed from the show menu. A log file can also be used. The rms errors for the training and the test data are plotted vs. iteration index in the training progress window.

This window can be kept open during network training.

Plot of training progress

The rms error for the training patterns and the test patterns (if any) are plotted vs. iteration index in the training progress window accessed from the show menu.

This window can be kept open during network training.

The **redraw interval** specifies the number of iterations between graph updating. The same update interval is used for the performance graph

Options menu

Copy graph - The graph can be copied to Windows clipboard as a metafile.

Multi-Layer Perceptron (MLP) Networks

The MLP algorithm implements multi-layer perceptron networks. Both feed-forward networks and partially recurrent networks with fictitious input and output nodes are implemented. Several alternatives for the node activation function are available. The network training is carried out using the Levenberg-Marquardt optimisation method.

Network size and configuration are set in the MLP setup window, accessed from the setup menu or by double-clicking the network picture.

Structure of the MLP networks

The network has an input layer, an output layer and may have internal hidden layers. Each layer has a number of nodes, also called neurons or elements. In the networks, the nodes in the input layer only distribute the input signals to the network. Each node in the hidden and output layers receive as input a weighted sum of the outputs, also called activations, of the nodes in the layer below. Each node in the hidden and output layers also has a bias term, which can be treated as a weight to a constant term with unity value. The weights and bias terms are parameters in the network training (optimisation). As an optional feature, direct (by-pass) connections between input and output nodes can be used.

Each node in the hidden and output layers has an activation function, which transfers the node input to an output signal. Several alternatives are available for the activation function, see MLP network configuration.

Limitations

The following limitations are set for the network size in NNDT:

- max 3 hidden layers
- max 15 nodes / layer
- max 200 parameters (weights, biases, initial states)

Network training, Levenberg-Marquardt method

The MLP networks are trained with the Levenberg-Marquardt method. The method minimises the squares of the residuals (=differences between desired outputs and network outputs) by modifying the network weights. The search direction is formed as an interpolation between the directions given by Gauss-Newton and the steepest descent methods. The λ (lambda) parameter, also displayed during run, determines the weighting of the two search methods. Steepest descent is dominating when λ is high, whereas Gauss-Newton is dominating when λ is low (near zero).

During and after training, the sum of the squared residuals (SSQ) and the rms error are shown in the main window. The rms error is given by

[illegible]

where *n* is the number of residuals.

MLP setup

The specification of the MLP networks are divided into a network configuration part and a training part. The specification is made in the network setup window, accessed from the setup menu.

MLP network configuration setup

MLP training setup

MLP network configuration setup

In additions to the configuration options specified below, the user can specify equal and/or constant weights. The network weights are assigned in the network state window.

Limitations

See MLP networks.

Number of nodes in each layer

The **number of nodes** can be set separately for each hidden layer. If the number of nodes is set to zero (-) for a layer, the layer will disappear. Note that the number of input and output nodes is specified in the pattern file setup window or by specifying recurrent connections.

Activation functions

For the nodes in each hidden layer and the output layer, different **activation functions** can be specified. Available node activation functions are:

-the (standard) sigmoid from 0 to 1

,

-the symmetric logarithmoid

,

-the linear (identity) function

,

-the sigmoid from -1 to 1

In the formulas above, *y* is the node output and *x* is the total node input.

By-pass connections

Direct **input-output** connections can be used to by-pass the internal network layers. If used, the weights for the by-pass connections are presented in additional lines in the network state window.

Feed-back connections

Recurrent links (feed-back links) can be specified from output nodes to fictitious input nodes. The links have a weight of unity and they are delayed with one sampling, i.e., the fictitious input nodes obtain the activations that the output nodes in question had at previous pattern. The links can be used to feed back true network output nodes, but also fictitious output nodes, that are in fact internal network nodes situated in the output layer. For each feed-back link specified, an additional fictitious input node is created. The initial state of the fictitious input nodes for each period are open parameters in the training (see MLP advanced setup)

MLP equal & constant weights

For certain problems, there may be a need to keep some of the network weights constant during training, or, keep the values of some weights equal. For a weight which is specified as constant, the initial value (read from setup file, or given by the user in network state window) is used throughout training.

The numbers (indexes) for the network weights are shown in a table (matrix) arranged similarly to the table in the network state window.

To specify that a weight shall be kept constant:

1. Select the weight number in the first dropdown list, or, click in the table.
2. Select "constant" in the second dropdown list.

To specify that two weights shall have equal values:

1. Select one of the weights in the first dropdown list, or, click in the table.
2. Select the other weight in the second dropdown list.

Several weights can be chosen by selecting an area in the table with the mouse.

The specified constants and equalities are shown in the table.

MLP training setup

In addition to the parameters listed below, more specialised options are chosen in MLP advanced setup window.

Data pre-treatment

If the patterns are to be used just like they are written in the pattern file, make sure that the box **Filter pattern data** is not checked.

A sliding window mean value filter is available, which filters the input and output signals in the pattern file. The user specifies a **number of old values in mean filter**, the filter is switched off by typing a zero in the box.

To reduce the size of the training data, each training pattern may be formed only after a given number of lines in the pattern file has been read (and filtered). The number is given as **number of samples between pattern formation**. Typing 1 in this box means that all lines in the pattern file are treated as different patterns.

The **number of patterns to be formed before training** parameter can be used to utilise only part of the pattern file.

Optimisation task

As alternatives to standard network training, where all network weights are parameters, gain factors for inputs and/or outputs can be specified to be the parameters in the **optimisation task**.

Maximum number of iterations

The **maximum number of iterations** specifies an upper limit for the number of steps in the training.

MLP advanced setup

Analytical derivatives

The Levenberg-Marquardt training method uses the derivatives of the residuals with respect to each weight in its search for optimal weight values. The derivatives are either obtained using analytical expressions or calculated numerically (as the change in the residuals for a small change in the weights). Analytical derivatives are more accurate and should normally give better results.

Limit parameter values

An upper and lower limit for the parameters (weights, biases and initial states) can be specified.

Limit relative changes

The maximum allowed relative change per iteration can also be specified. The step suggested by the training method is restricted if any parameter would change more than the maximum percentage specified.

Penalty factor

A penalty term can be specified to keep the weights at reasonable sizes without specifying definite limits. The penalty term is treated as an additional residual which is minimised along with the output residuals in the training. The term is given by

[illegible] $| \; |$,

where γ is the factor specified by the user and *w* denotes the network parameters (weights, biases and initial states).

Number of separate periods in training data

For network with feed-back links (i.e. recurrent networks), the initial states of the fictitious input nodes are open parameters, just like the network weights. To enable the use of training data consisting of several different parts, the **number of separate periods** and the locations of the **first patterns in the periods** can be specified. When a new period starts, the fictitious input nodes are set to their initial activations. The initial states may be either **equal** for all periods or specific for each. Note that the start of each period is given as the pattern number in training. Depending on the data pre-treatment chosen, this number does not necessarily equal the pattern number in the pattern data file.

Teacher forcing

For networks with feed-back links from true output nodes, teacher forcing may enhance the training performance. Teacher forcing means that, instead of network outputs, the desired output signals are fed back to the fictitious input nodes. The user specifies the interval between teacher forcing actions; typing 1 in the box means that desired (target) output values are fed back at each pattern. Teacher forcing is never used for the first pattern in a period.