

POWER SYSTEMS, SUCH AS LOAD FORECASTING, FAULT DIAGNOSIS, ALARM HANDLING AND STATE ESTIMATION WITH MULTIPLE NEURAL NETWORK

PRUTEANU EUSEBIU, COSTICA NITU, ALEXANDRU DANIEL ENESCU

University of Bacau, University "Politehnica" of Bucharest

Abstract: Artificial Neural Networks proved to be capable of finding internal representations of interdependencies within raw data not explicitly given or even known by human experts. This typical characteristic together with the simplicity of building and training Neural Networks and their very short prediction delay time encourages researchers to apply neural networks to different tasks for power systems, such as load forecasting, fault diagnosis, alarm handling and state estimation. By combining more than one of these neural networks, better performance can be achieved with these new architectures.

Key words: multiple neural network, power system, network architecture.

1. INTRODUCTION

Neural network can provide good results in short time scales – but only for certain types of problem. Describing the technology as “computers that think” leads to unrealistic expectations and discredits an important and valuable engineering tool. Neural networks should be considered as components within an overall application – not as solutions in their own right. Typically, they are used in conjunction with both conventional processing techniques and rule-based processing (such as expert systems technology). Neural computing systems have a number of attributes that can be used with advantage in applications:

- learning from experience;
- generalising from examples;
- extracting essential information from noisy data;
- developing solutions faster and with less reliance on domain expertise;
- adaptability;
- computational efficiency;
- non-linearity.

2. MANAGING APPLICATION DEVELOPMENTS

Neural computing projects differ from many conventional software development projects for a number of reasons:

- Projects are data driven. Therefore, there is a need to collect and analyse data as part of the design process and to train the neural network. This task is often time-consuming and the effort, resources and time required are frequently underestimated;

- It is not usually possible to specify fully the solution at the design stage. Therefore, it is necessary to build prototypes and experiment with them in order to resolve design issues. This iterative development process can be difficult to control;
- Performance, rather than speed of processing, is the key issue. More attention must be paid to performance issues during the requirements analysis, design and test phases. Furthermore, demonstrating that the performance meets the requirements can be particularly difficult.

These issues will affect the following areas of management:

- project planning;
- project management;
- configuration management;
- documentation.

Typical neural computing project lifecycle has three main phases:

- Application identification and feasibility – the objectives of this phase are the same as for any IT projects, namely to identify applications and establish the feasibility and business case for the proposed development.
- Development and validation of prototype – is concerned with designing, developing and validating a prototype of the final application. This method is a well established technique for conventional IT projects, and is particularly appropriate for neural computing projects.
- Conversion of prototype into deliverable system, delivery and maintenance – is concerned with converting the prototype into operational software and, possibly, hardware, and implementing any additional functions (e.g. specific interfaces) that it requires.

Project planning and control can be seen as a cyclic activity with three stages: plan, act and review. Monitoring and control activities for neural computing projects are the same as with conventional projects: only the prototyping stage is different. This stage needs more detailed monitoring to determine when to terminate each of the iteration of prototyping. There are two situations in which it will be necessary to conclude an individual prototyping cycle earlier than planned:

- when the goals for the prototyping cycle have been achieved early;
- when the current work is unlikely to give satisfactory results.

Usually, the goals will be expressed in terms of performance criteria for the neural network under development, and their achievement will be a key factor in determining when to stop the work on a prototype cycle.

Reviewing progress is a key part of an IT project management method, and neural computing projects are no different. However, there are three types of reviews that are particular to neural computing projects:

- Data reviews. The aim is to gain an understanding of the data and to determine whether it is sufficient for the requirements of the project.
- Reviews at the end of each prototyping cycle. The results of this review will form the basis for planning the next cycle of prototyping.
- Overall review at end of the prototype development phase. The review should concentrate on determining the optimum technical solution.

Configuration management in neural network design is needed just as on a conventional computing systems development project. The prototyping phase will generate a large number of experimental results that will be used to determine the final system. The aim of the configuration management system must be to ensure that these results are recorded.

Documentation is a vital part of the design process: it is the means both of recording the design process and communicating the final design to those who will implement it. The design documentation must record information relevant to neural computing developments, such as:

- details of pre-processing algorithms;
- neural network input variable scaling;
- neural network architecture;
- neural network size;
- details of training algorithms, parameters, stopping method and training conditions;

- achieved performance.

3. MANAGING MULTIPLE NEURAL NETWORK ARCHITECTURES

Sometimes, better performance can be achieved by combining more than one type of neural network in a system, requiring more detailed problem analysis. The first step is to decide how many specialised neural networks are required. This depends on the structure of the problem and the results of the data analysis, which should indicate the number of sub-problems or data regimes. After deciding how many neural networks to use, choosing the size of each can be determined following usual guidelines. Here are a few examples:

- In prediction problems where the time-series is strongly cycling with a known period (e.g. hourly predictions over a daily cycle), a simple approach would use an MLP where one of the inputs represented time of day. Since the behaviour of the time series will often be different at different times of day, an alternative solution would be to use multiple neural networks to cover these periods, i.e. different networks for different times of the day;
- A second example is the “supervisor neural network architecture”. This is a hierarchical structure of neural networks where a problem is divided into several sub-problems and a neural network is associated with each. A supervisor, which can be a conventional process or another neural network, adjudicates between the outputs of the multiple neural networks to determine the correct output. This structure has proved to be very effective in complex pattern recognition problems, where a single large MLP was unable to generate a sufficiently complex model of the data.

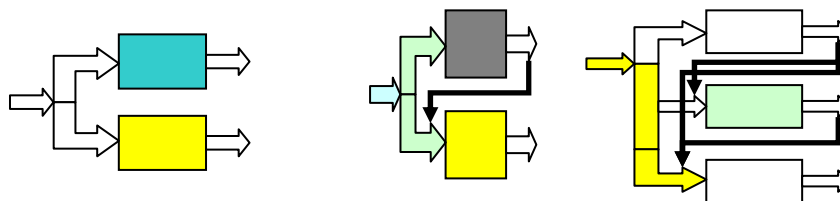


Fig.1. Multiple Neural Network Architectures

4. IDENTIFYING APPLICATIONS

Neural computing requires a number of technical features that are not normally a feature of conventional computing. If a candidate application does not possess these features, then neural computing may not be an appropriate solution.

The features which indicate applications where neural computing may give an advantage over other technologies are:

- The application deals with poor quality or incomplete data;
- The application requires integration of different types of input data, e.g. a combination of computer data and signals from sensors;
- It is difficult to specify a model for mathematical simulation, or rules for a knowledge based system;
- The application needs to be adaptive – i.e. the neural network must be capable of learning during operation, adapting its responses as the operating environment slowly changes.

There are also a number of practical issues that should be checked:

- Are adequate resources available? These resources must be sufficient to deal with the significant learning curve associated with a venture into a new technology.
- Consider any objections that people may raise to the new technology. If the reliability of neural computing cannot be given, the application may not be used.
- There may be considerable practical problems associated with the collection of the data:
 - If training data has not yet been collected and compiled in a computer readable form, it will be necessary to collect data as part of the project;

- Special instrumentation and recording facilities may be required, and specific experiments may be needed to ensure that the data covers the necessary range of conditions;
- Storing large quantities of data can itself constitute a significant problem.

5. COLLECTING AND PREPARING DATA

The collection of data is an important early step in the implementation of neural computing application developments. Data is absolutely crucial for developing neural computing applications, in particular for design, training, testing and validation. A clear, documented, plan of action is strongly recommended to:

- Minimise disruption to the organisation;
- Control data collection costs;
- Ensure smooth operation of data collection;
- Ensure data for the project is available at the right time.

The data collection plan typically consists of four tasks:

- Identify the data requirements - what data the project needs;
- Identify the data sources – where the data will come from;
- Estimate the data quantity – how much data will be needed;
- Specify and agree the data format – how the data will be collected and stored.

When the raw data has been collected in the agreed form, it may need converting into a more suitable format. At this stage, must:

- Perform data validity checks. Data validity checks will reveal any patently unacceptable data that, if retained for neural network training, would produce poor results.
- Assess the data set quality. As well as checking the validity of the data, it is equally important to ensure that the overall distribution of points within the training set is not flawed.
- Formulate a strategy for dealing with poor quality data. There are two ways of making allowance for poor data: use the data “as is” (but reduce the significance the neural network gives it by modifying the associated internal weights during training) or attempt to correct whatever makes the data invalid, and then treat it as normal, acceptable quality data.
- Partition the data (the process of dividing the data into the training, test and validation tests). The primary concerns should be to ensure that:
 - The training set contains enough data, and a suitable data distribution, to adequately demonstrate the properties that neural network must learn.
 - There is no unwarranted similarity between data in different data sets.
- Document the data.

6. OPTIMISING AND VALIDATING THE PROTOTYPE

It is possible that the cost function has particularly high values, even at the end of training when it has stopped decreasing (high cost function values at the start of training are not a problem). This problem, known as “high final training error”, is caused by some aspect of the neural network’s design being unsuited to the application.

Prototype testing tests the neural component only. Prototype validation, on the other hand, validates the whole system including the non neural components. Sometimes, validation may lead to conclude that it is necessary to do more optimising before developing the delivery system. This will require more training and testing and further validation.

Validation involves the presentation of data that was not used at any stage of training and testing. If the validation set contains values outside the nominal range of operation, it can also be used to establish the bounds of validity of the neural network and what its behaviour will be when these bounds are exceeded. Information about the boundary performance is valuable because the user can be alerted to the fact that neural systems performance breaks down in some circumstances.

7. APPLICATIONS OF MULTIPLE NEURAL NETWORKS IN POWER SYSTEMS

7.1. Load forecasting in power systems with neural networks.

This method allows the load prognosis of the system in the next 24 hours (/2 hours/1 hour).

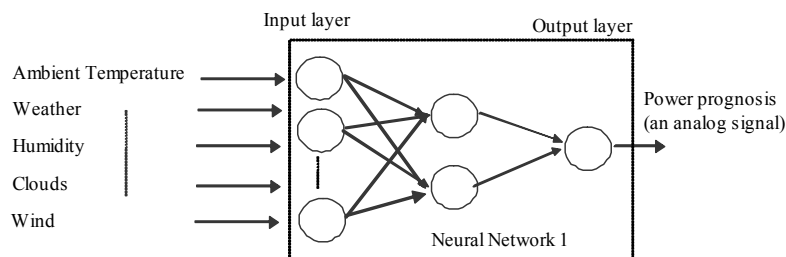


Fig.2. Neural Network for load forecasting

7.2. Fault detection in power and distribution systems.

Fault diagnosis concerns the monitoring of power system component. The method needs measurement of hundreds of parameters: voltages, currents etc. Sensors are placed at several spots in and on the object to be monitored. Using this method, the generated alarm signal is a logical one.

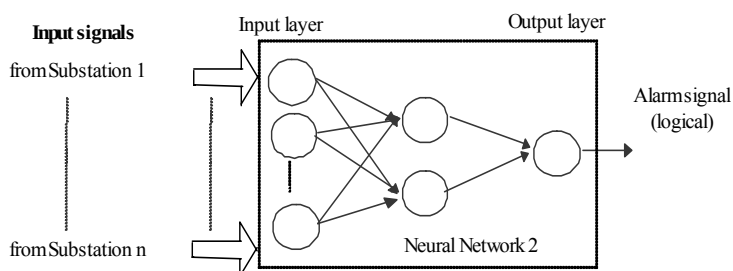


Fig.3. Neural Network for fault detection

7.3. Power system security analysis.

The main task in power system security analysis is to decide whether the system is currently operating in the normal state where all constraints on bus voltages, active and reactive bus power, line power flows and frequencies are respected, and whether the system stays stable with respect to voltage and frequencies. Using a neural network, the signal obtained represents an indicator which can tell if the system operates (or not) in normal state. The state vector shows the stability of the power system and the parameters of the transitory processes.

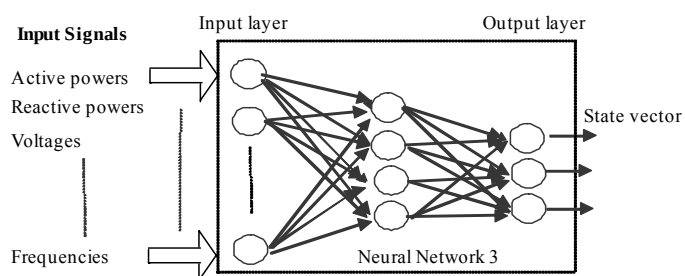


Fig.4. Neural Network for security analysis

7.4. Distribution system state estimation.

Using neural networks, the state estimation intends to predict the violation of constraints for voltages, power flows,, voltages drops and islanding in case of an outage.

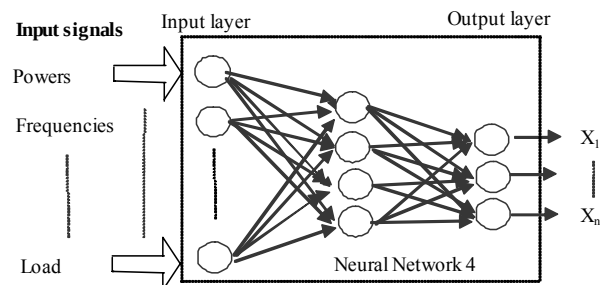


Fig.5. Neural Network for state estimation

The presented neural networks used for applications in power systems can be interconnected in different structures as discussed in section 3.

8. CONCLUSIONS

Electrical power systems represent complex systems involving many electrical components whose operation has to be planned, analyzed, monitored and controlled. The time-scale of tasks in electric power systems extends from long term planning years ahead to milliseconds in the area of control. The behaviour of power systems is highly non-linear. Monitoring and control involve several hundred variables which are only partly available by measurements. Artificial neural networks (ANN) are designed to handle non-linear tasks. Their main application areas are in the domain of pattern recognition, classification, interpolation and approximation.

REFERENCES

- [1] Nitu C., Krapivin V.K., Bruno A., *Process Modelling in Ecology*, Ed. Printech, Bucuresti, 2000;
- [2] Costache G., *Efficient Control Method for Nonlinear Systems*, The 12'th Int. Symposium on Modelling, Simulation and Systems' Identification, Galați, September, 2004, p. 65-71;
- [3]. C.Nitu, V.F.Krapivin, *Intelligent technics in ecoinformation systems*. Int.Conference CSCS12, v.2, pp.74-79, Bucharest, 26-29 May, 1999