December, 1992

Possible Applications of Neural Nets to Power System Operation and Control

P. K. Kalra
Alok Srivastava
D. K. Chaturvedi, Dayalbagh Educational Institute

Available at: https://works.bepress.com/dk_chaturvedi/3/
Possible applications of neural nets to power system operation and control

P. K. Kalra and Alok Srivastava  
Department of Electrical Engineering, Indian Institute of Technology, Kanpur 208 016 (India)  
D. K. Chaturvedi  
Faculty of Engineering, Dayalbagh Educational Institute, Dayalbagh, Agra 282 005 (India)  
(Received May 6, 1991)

Abstract

Problems related to power system operation and control are complex and time consuming because of the non-linearities involved in their formulation and solution. Fast solutions to these problems can be obtained only through parallel processing. Neural nets provide massive parallel processing facilities and may also be used efficiently to model systems with non-linearities. The capabilities of neural nets can, therefore, be well utilized in modelling and processing problems related to power systems. In order to reduce the burden on computers, algorithms involving optimization and complex equations can be converted to heuristics. These heuristics can then be represented in terms of rules and an expert system can be built, with the added advantage of obtaining solutions in a time intensive fashion. This paper studies the application of neural nets to problem solving in power system operation and control, and demonstrates how present methods for solving such problems can be converted to the neural net approach.

Introduction

The early 1980s saw the development of expert/artificial intelligence/knowledge based systems for power systems [1–10] and later on the application of neural nets to power systems [11–15]. These approaches attempted to model the thinking and processing procedures of the human brain and utilized the number crunching capabilities of computers. They also used non-algorithmic methodologies so as to reduce the influence of approximations of mathematical modelling to a minimum. Work done in pattern matching played an important role in implementing these approaches [16–20]. Improvements were steadily made in decision-making procedures in the expert system domain for handling uncertainties and incompleteness in available knowledge through inductive and fuzzy reasoning.

Basic principles of neural nets

Neural nets attempt to simulate the thinking and processing procedures of the human brain by modelling the neuron. The basic component of a neuron is known as the soma, which is attached to the axon (Fig. 1). The axon is electrically active and produces the pulse emitted by the neuron. Electrically passive dendrites receive input from neurons by means of specialized contacts called synapses, which act as weights to the input information. The neuron is fired only when the sum of the weighted inputs is above a certain
threshold. Information received by neurons may be processed in parallel or sequentially or as a combination of both. For some kinds of stimuli, the reaction of the brain is typecast in the sense that, for a certain input, only a specific output is obtained. Although the mechanism behind this process is not fully understood, research has started on the simulation of processes in neural nets which can match inputs to required outputs and incorporate variations in input patterns to account for output patterns. This pattern matching ability of neural nets has played a vital role in their implementation.

Human behaviour is the integration of hierarchical, modular, distributed computation and automatic learning processes. The learning process is based on declarative and reflexive mechanisms. In the declarative state an assessment is made of why or how a certain action is to be performed. In the reflexive state, a reaction appears to the action. Tasks learnt from the declarative state normally become reflexive over a period of time. Neural nets simulate such processes by training.

Neurons interact in feedforward, feedback, fully connected or partially connected fashions. The connections of the neurons in the brain as well as in the neural network model are shown in Fig. 2. The feedback path influences the nature of adaptivity and trainability. If the selection of weights and thresholds in an artificial neural net is automated, then this could be thought of as a learning mechanism. This learning capability of neural nets distinguishes them from conventional computer software. At present, neural nets show good potential for ever-improving performance through dynamic learning.

A single neuron in a neural net can be represented as in Fig. 1, where $X_i$ is the input, $W_i$ is the weight carried by the input $X_i$, and box $A$ represents the linear combination of weighted inputs.

The output $Y$ of box $A$ is passed through a non-linear function called the sigmoid or learning function (Fig. 3). For nerve connections in the representation of Fig. 1 to be a good approximation to an actual neuron, all neurons connected together must form a stable system. The functionality of this system can be found by modifying the weights. One can think of the combination of neurons and synapses as hardware while weights and thresholds can be treated as software.

Neural nets can be viewed as one of the following: (i) a set of non-linear differential equations or non-linear difference equations, or (ii) a non-linear transformation between input and output.

Information processing in neural nets is non-algorithmic, so the influence of approximations of mathematical modelling is reduced to a minimum. Further, neural nets are capable of handling uncertainties so that results obtained through trained neural nets, even with partial inputs, may be very close to the results with complete knowledge. Depending on the neuron interaction, neural nets can be classified as: (a) feedforward; (b) feedback.

(a) Feedforward neural networks. Here, neurons are arranged in layers like directed graphs. Inputs are applied to the layers and outputs are collected. Stability is not a problem here because these networks are loop free. The computation time in such neural nets is the time required for signals to propagate and the output to settle.

(b) Feedback neural networks. Here, neurons are arranged in the form of undirected graphs. The connections in this case are symmetrical and bidirectional. Feedback neural network models are sequential or asynchronous. Here the system is initialized and then it evolves to a final state.
in the course of time. The stability of this type of neural net is analysed with the help of energy functions that can be defined in terms of states or neurons, weights and thresholds. Under certain conditions, the energy function decreases monotonically as the network moves from one state to another. The stability and convergence can be analysed by studying the descent of scalar energy functions instead of the transition of states.

**Neural nets in power systems**

Power systems may be viewed as a combination of the following activities.

(i) **Planning.** To operate the system efficiently and effectively, the load has to be forecasted, load flow has to be performed, and short-circuit currents have to be evaluated so as to have steady-state, dynamic, and transient stabilities. Most of these studies are performed off-line.

(ii) **Design.** Ground electrodes have to be selected and equipment such as transformers designed.

(iii) **Control.** Closed-loop and open-loop controls have to be used for fuel control, load frequency control, excitation control, voltage control, and high voltage direct current (HVDC) system control.

(iv) **Operation.** Fault diagnosis, contingency analysis, security assessment, and state estimations, etc. must be performed on a short-duration basis. These studies are done in real time/on-line.

Traditionally, a power system operates with large uncertainties. Retrieval of complete knowledge of the interconnections between variables to be controlled is very complex because of the size and non-linear behaviour of the system.

In the existing system, a decision taken in the case of changes in operating conditions involves the following:

1. use of a large data bank,
2. conversion of data to usable information,
3. use of input from past experience,
4. implementation of control tasks.

The last approach is considered to be the most effective. The decision-making procedure can be thought of as a pattern recognition problem.

**Problems which can be solved using neural nets**

In power systems, stability assessment and security monitoring problems can be solved by using pattern matching techniques. Depending upon the class of pattern, an appropriate action can be taken. Neural net based decision procedures are capable of recognizing patterns and responding in a real-time manner, and these problems are, therefore, very good candidates for their application. Another class of problems is load forecasting, state estimation, adaptive control, and bad-data detection. These are solved by using difference equations, differential equations or recursive curve fitting. Since neural nets themselves are defined in these terms, they are very good tools for solving such equations. Further, data may be polluted by noise, incomplete or widely varying in nature. Neural nets, therefore, provide a particularly effective solution as they are capable of handling uncertainties.

A third class of problems includes load flow analysis, contingency analysis, prediction of voltage collapse, and emergency control actions. These problems are difficult to solve because the mathematical equations involved and the optimization of their solutions is time consuming. By applying neural nets to these problems and training them extensively, it may be possible to reduce the size of data banks and knowledge bases with the added advantage of real-time solutions.

Neural nets can also be applied to the design of controllers for automatic gain control, excitation systems, HVDC systems and static VAR systems for large changes in operating conditions. Designers of these controllers could encounter the following problems:

1. redundancy in data used for system modelling;
2. limitations in implementation of control laws based on heuristics or complicated mathematics (fuzzy theory or expert system based approaches, too, face limitations in terms of flexibility);
3. restrictions in measurements of system parameters.

A standard control system tries to minimize the error between the reference and measured signals. In control systems associated with power systems, this minimization procedure is rendered more complex because of the non-linearities involved, and because of the multivariable or multiple input and multiple output (MIMO) nature of the system. To achieve fast, reliable and optimal performance of control systems, control loops can be implemented through parallel computer architecture. The problem of non-linearities can be partially solved by using intelligent
control systems. Complexities such as time dependence can be solved only through large knowledge based expert systems. These, however, have inherent drawbacks, as learning procedures are neither fully understood nor possible to implement completely. Further, search processes may be intensive, hence time consuming. Neural nets, on the other hand, can handle situations of incomplete information, corrupt data, and large data volumes, thus providing control systems with close to ideal performance. While the training period may be lengthy, once trained, neural nets can be used for real-time applications as they do not employ any search techniques.

Neural nets have already been applied to the following problems in power systems: location of harmonic sources, distribution systems, location of shunt capacitors in radial systems, and load forecasting.

Illustration: load forecasting using neural nets

A neural network model has been designed and tested at the Indian Institute of Technology, Kanpur, for short-term load forecasting. The details of the neural network model and that of the training algorithm used to achieve this goal are given below.

Network structure (Fig. 4)

The artificial neural net (ANN) used in this problem is a two-layer feedforward net. The two layers are the hidden layer and the output layer. Input values are fed directly to the input layer which simply distributes them among different hidden layer connections. An auxiliary node has been inserted to decide the threshold of the nodes in the network and the input to these nodes is a biased value (1.0 in the present implementation).

Training algorithm and parameters

A standard back-propagation algorithm [19] was used to train the feedforward type of neural network. The algorithm is summarized in the steps given below. The mathematical formulations (A), (B), (C), (D), (E) and (T) are described later.

Step 1. Set the initial weights and thresholds for all inputs using random numbers between 0.0 and 1.0 and set the number of iterations $N$ to 0.

Step 2. Read inputs and desired output in appropriate form (A).

Step 3. Find the output of each layer and then the final output (B) using the threshold function (T) and increment the number of iterations.

Step 4. Find the error $E$, that is, the deviation (C) from the desired output. If $E > E_0$ or the number of iterations $N > N_0$, stop processing and repeat the loop a fixed number of times to ensure better learning; else go to step 5 (here $E_0$ is the maximum permissible error and $N_0$ is the maximum number of iterations permitted).

Step 5. Calculate error functions $\delta_i$ for the output layer and other hidden layers using (D).

Step 6. Modify the weights and thresholds using (E). Go to step 3.

These operations are to be performed for every set of input data over which we want to train the neural network. When the training with the first set of inputs is over, we present the new set and desired output for that input set and so on. After all the input sets are entered, the training is repeated with fewer iterations to improve the performance and the fault tolerance of the neural network model.

The mathematical formulations used above are as follows.

(T) Threshold function: sigmoid function

$$f(x) = \frac{1}{1 + \exp[-(x-x_0)/\theta]}$$

where $x$ is the variable value, $x_0$ the threshold, and $\theta$ the slope of the sigmoid function.
For finite $x_0$,
\[ f(x) = 0 \text{ as } x \to \text{ negative infinity} \]
\[ = 1 \text{ as } x \to \text{ positive infinity} \]
\[ = \text{ some value between 0 and 1 for any other } x \]

In the load forecasting problem we used:
\[ \theta = 1 \]
\[ x_0 \text{ as obtained by training} \]

(A) The inputs are taken in normalized form between 0.1 and 0.9 using the formula
\[ x = \frac{x - \text{MIN}}{\text{MAX} - \text{MIN}} \times 0.8 + 0.1 \]

MAX and MIN are the global maximum and minimum, respectively, in the input set.

(B) The output of any layer $j$ is defined as
\[ o_j = f\left( \sum_i W_{ij}x_i - x_0 \right) \]
where $f(\cdot)$ is the threshold function described above, $W_{ij}$ is the weight between node $i$ of the input and node $j$ of the next layer, $x_i$ is the input to node $i$, and $x_0$ is the threshold set by the auxiliary node connection between the auxiliary node and the node in which we are interested.

(C) The mean error criterion is used to measure the deviation of the neural network’s output from the desired output:
\[ \text{error} = \frac{1}{2} \sum_i (\text{output}_i - \text{desired output}_i)^2 \]

(D) Error functions
The error for the output layer: for the $k$th node in the output layer the error function used is $\delta_k$, which is represented as
\[ \delta_k = (t_k - o_k) o_k (1 - o_k) \]
where $o_k$ is the actual output and $t_k$ the desired output.

The error for the other layers: since the output of the hidden layer is not known, a different error function is used to evaluate the error for each hidden layer node as given below:
\[ \delta_j = o_j (1 - o_j) \sum_k W_{jk} \delta_k \]
where $o_j$ is the output of node $j$, $\delta_k$ the error of node $k$ in the output layer, and $W_{jk}$ the weight between node $j$ and node $k$ as described above.

(E) The weights of the neural net were modified using the error functions and the weight change as follows:
\[ W_{ij}(\text{new}) = W_{ij}(\text{old}) + \eta \delta_j i_i + \alpha [W_{ij}(\text{old}) - W_{ij}(\text{\star})] \]
where $\eta$ and $\alpha$ are acceleration functions and $W_{ij}(\text{\star})$ is the value of the weight two steps before the current one.

Application to load forecasting
In this application a neural network is trained with the past four hours of data to forecast the load for the fifth hour using a feedforward network trained for forecasting with the back-propagation algorithm. The implementation details adopted for the load forecasting are given below.

Input set
Data were taken from Monday to Thursday. We took three test hours, the 5th, the 11th and the 18th. For each hour, four previous hours of data (normalized as described above) were taken. Table 1 shows the input values used for training for four different days of a week. The input value consists of the normalized load data for five hours per day, out of which four normalized load values are taken as input data and the fifth one as the desired output of the neural net.

Training
The following parameters were selected for training the neural network model for load forecasting.

The slope of the threshold function $\theta$ was taken as unity for training.

Table 1. Input set used to train the ANN in the short-term load forecasting application

<table>
<thead>
<tr>
<th>INPUTS FOR THE NEURAL NETWORK</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUTS FOR DAY 1</td>
</tr>
<tr>
<td>INF. 1: 1.7000000000E-01</td>
</tr>
<tr>
<td>INF. 2: 1.2000000000E-02</td>
</tr>
<tr>
<td>INF. 3: 1.1000000000E-02</td>
</tr>
<tr>
<td>INF. 4: 1.1400000000E-01</td>
</tr>
<tr>
<td>D. OP.: 1.4000000000E-01</td>
</tr>
<tr>
<td>INPUTS FOR DAY 2</td>
</tr>
<tr>
<td>INF. 1: 1.6000000000E-01</td>
</tr>
<tr>
<td>INF. 2: 1.3000000000E-01</td>
</tr>
<tr>
<td>INF. 3: 1.1000000000E-01</td>
</tr>
<tr>
<td>INF. 4: 1.1200000000E-01</td>
</tr>
<tr>
<td>D. OP.: 1.3000000000E-01</td>
</tr>
<tr>
<td>INPUTS FOR DAY 3</td>
</tr>
<tr>
<td>INF. 1: 1.8000000000E-01</td>
</tr>
<tr>
<td>INF. 2: 1.4000000000E-01</td>
</tr>
<tr>
<td>INF. 3: 1.3000000000E-01</td>
</tr>
<tr>
<td>INF. 4: 1.4000000000E-01</td>
</tr>
<tr>
<td>D. OP.: 1.6000000000E-01</td>
</tr>
<tr>
<td>INPUTS FOR DAY 4</td>
</tr>
<tr>
<td>INF. 1: 1.7000000000E-01</td>
</tr>
<tr>
<td>INF. 2: 1.1000000000E-01</td>
</tr>
<tr>
<td>INF. 3: 1.2000000000E-01</td>
</tr>
<tr>
<td>INF. 4: 1.2400000000E-01</td>
</tr>
<tr>
<td>D. OP.: 1.4900000000E-01</td>
</tr>
</tbody>
</table>

D. OP.: DESIRED OUTPUT

* THE VALUES GIVEN HERE HAVE BEEN NORMALISED BETWEEN 0.0 AND 1.0. 
The accelerations and momenta were as follows:

for the initial training,

$$\eta = 0.9 \quad \text{and} \quad \alpha = 0.05$$

for the later part due to large weight variations,

$$\eta = 0.3 \quad \text{and} \quad \alpha = 0.8$$

Table 2 shows the weights that were generated after the training of the neural net.

### Outputs

Table 3 shows the results obtained from the trained network. The output has been predicted for day 5, that is, Saturday, about which the network was given no information. The output obtained is within an error of 1.22% (max). The limiting error and maximum number of iterations $E_0$ and $N_0$, respectively, were taken as $E_0 = 0.0001$ and $N_0 = 5000$.

### Conclusion

In this particular application, the neural network predicted the loads for three different hours on the same day. Round-the-clock predictions can be made in a similar fashion. This neural network as such does not model holiday loads, but they can be forcast with a separate network of a similar kind but with a different training. To make this neural net more general and foolproof, extra parameters such as temperature, humidity, etc. can be helpful while training and during prediction. Using these parameters it may be possible to predict the load for any month.
or season of the year by using just one neural net model.

**Future of neural nets**

Neural nets can be applied to a wide variety of problems in power system operation and control but more thought needs to be given to some aspects, such as:

1. improvement of the efficiency of the learning procedures, as the back-propagation approach is slow;
2. maintenance of software and hardware in the control rooms of substations;
3. comparison of various neural net structures;
4. the design of new threshold function structures which simulate more accurately the nonlinearities involved.

Knowledge based systems can be used together with neural nets to eliminate the shifts between declarative and reflexive mechanisms. One example which can be cited is the location of capacitors using artificial intelligence (AI) methods. Once the capacitors have been located, neural nets can be used for selecting the steps to put them into service. Further research, therefore, must address the integration of AI based systems with neural nets.

Another problem is to coordinate HVDC controllers for multiterminal or point-to-point systems by switching over among controllers. This problem can be resolved through a dual approach: (i) performance of tasks while the neural net learns (AI based system); (ii) learning through experience to improve system performance (neural net).

Research problems in the domain of integration of AI based systems and neural nets may address the following:

1. the possibility of exchange of knowledge between an AI based system and neural nets;
2. a knowledge based system learning from neural net performance;
3. division of knowledge between neural nets and knowledge based systems;
4. use of symbolic computation in AI for neural nets;
5. creation of learning rules for AI based systems from neural nets and vice versa.

**Concluding remarks**

This paper brings to the reader the background and thinking needed by power system engineers for application of neural nets to power system operation and control. At the Indian Institute of Technology, Kanpur, the problems of short-term load forecasting, capacitor location, controller design and prediction of AC voltage instability for AC–DC–AC systems have been taken up for the application of neural nets. Our experience has been very good, with some measure of success. A neural net simulator will soon be ready. It is based on a multilayer feedforward network, with a back-propagation scheme of learning.

In the opinion of the authors, neural nets will have a great impact on electric power system operation by providing fault tolerance and massive parallel and distributed processing.

**References**


