ELSEVIER

Electric Power Systems Research 34 (1995) 127–134

ELECTRIC POWER SYSTEMS RESEARCH

A neural network preestimation filter for bad-data detection and identification in power system state estimation

H. Salehfar, R. Zhao

Department of Electrical and Computer Engineering, Clarkson University, Potsdam, NY 13699-5720, USA

Received 7 February 1995

Abstract

Popular state estimation techniques in industry are mostly based on the weighted least squares (WLS) method and its derivatives. These estimators usually detect and identify multiple gross measurement errors by repeating a cycle of estimation–detection–elimination. It is rather time consuming for large systems. This paper presents a neural network preestimation filter to identify most forms of gross errors, including conforming bad data, in raw measurements before state estimation rather than afterwards. The proposed neural network model is trained to be a measurement estimator by using the correct measurements of typical system operating states. Once trained, the filter quickly identifies most forms of gross measurement errors simultaneously by comparing the square difference of the raw measurements and their corresponding estimated values with some given thresholds. System observability is maintained by replacing bad data with their reasonably accurate estimates. Using the proposed neural network preestimation filter, the efficiency of present state estimators is greatly improved. Results from several case studies are presented.

Keywords: State estimation; Data processing; Neural networks; Back-propagation algorithms

1. Introduction

State estimators are an important component of energy control centers [1,2]. System measurements acquired by control centers are processed by a state estimator to determine the state of the system and to provide a reliable system database. If one or more of the telemetered measurements are contaminated with gross errors, the estimated system state and the corresponding database will be biased. In order to avoid this problem, several powerful bad-data detection and identification methods have been developed and reported in the literature [3–12]. Most common state estimation techniques in industry are based on the weighted least squares (WLS) method. The WLS estimators detect and identify single and multiple gross measurement errors by testing the normalized residuals of the measurements. These state estimators usually repeat a cycle of estimation–detection–elimination until an acceptable result is obtained. For large systems, it is rather time consuming to perform such bad-data detection and identification procedures online. Furthermore, the WLS based estimators cannot effectively detect and identify multiple interactive and conforming bad data.

This paper presents a preestimation bad-data detection and identification method based on back-propagation neural networks. The objective of this neural network preestimation filter is to detect and identify gross errors in raw measurements before state estimation rather than afterwards with the heavy tools of statistical analysis. Some type of filtering of gross measurement errors is currently being used. However, the filters are very elementary and have a limited useful range of operation. Unlike traditional computing methods, neural networks synthesize a mapping between input and output variables by learning from a set of training examples. In the proposed method, neural networks are trained by a large number of typical system measurement patterns which cover the entire space of load demand and several feasible changes in the system topology. Due to its parallel distributed processing nature, the proposed model, once trained, can quickly and rapidly detect and identify most forms of gross errors including multiple interactive and conforming bad data simultaneously. Once bad data are identified, the proposed technique can automatically replace them with their reasonably accurate estimates to avoid system unobservability problems. It is expected that the proposed model will greatly improve the efficiency and speed of present state estimators.

SSDI 0378-7796(95)00966-L

The organization of this paper is as follows. In the second section, the fundamentals of our proposed method are discussed. The model is applied to the IEEE 30-bus system and the results are discussed in the third section. Finally, in the fourth section of the paper, the concluding remarks are presented.

2. Method formulation

Artificial neural networks (ANNs) have been developed to match human's powerful thinking, remembering, and problem-solving capabilities. Due to their capability to map complicated and nonlinear relationships of input–output patterns, ANNs are found to be a powerful computing technique, and have thus inspired many people to apply them to various system problems [13–17]. Most applications of ANNs show that the back-propagation technique is a very powerful tool for constructing nonlinear transfer functions among several continuous-valued inputs and one or more continuous-valued outputs. A good description of the back-propagation technique is given in Ref. [18].

In this paper, the proposed preestimation filter detects and identifies gross errors in raw measurements before state estimation, as shown in Fig. 1. The back-propagation neural network is used as a measurement estimator, and is designed to receive the raw measurements and provide their estimated values. Bad data are then flagged if the difference between the measured and estimated values of the measurement variables is larger than a given identification threshold. During the training stage, the output part of a training pattern is formed exactly the same as the input part, and both parts consist of correct measurements of a typical system operating state. To generate a complete set of training examples, a large number of typical system operating states are collected and simulated by running several standard power flow studies. When the neural network is being trained, the connection weights are corrected to minimize the error between the true and estimated values of the measurement variables. In a neural network, the weights are the distributed associated memory units and represent the current state of knowledge. Measurements from a system operating state in the training examples are represented by all weights and they share weights with the measurements

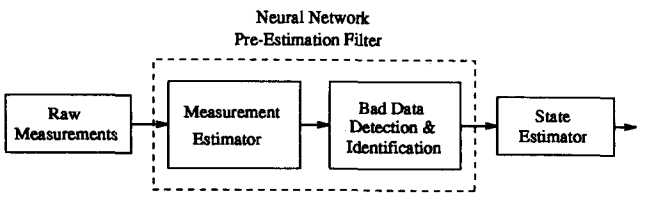

Fig. 1. Preestimation filter.

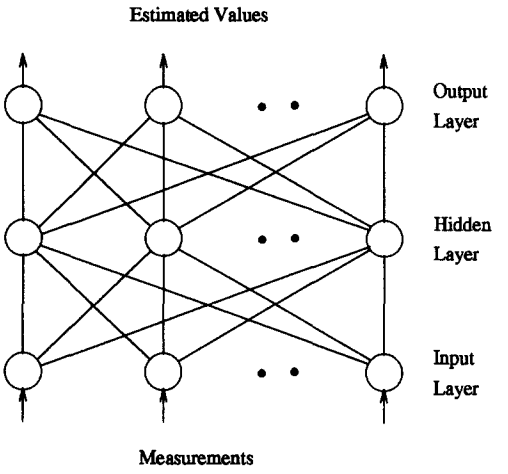

Fig. 2. Architecture of the neural network measurement estimator.

from other system operating states. This will lead to a reasonable network response when the network is presented with incomplete, noisy, or previously unseen inputs, and is referred to as 'generalization'. In other words, whenever a trained neural network is presented with corrupted raw measurements, the network will look upon bad data as noise and will choose the closest match in memory to these inputs to generate the 'intelligent' outputs. These outputs are the estimated values of raw measurements, in which bad data are replaced with their estimated values closer to their true values.

2.1. Architecture of the measurement estimator

Generally, a back-propagation neural network consists of many processing units, also known as neurons, organized into an input, an output, and at least one hidden layer. Each layer is connected to the succeeding layer through the connection weights. The architecture of the back-propagation neural network chosen for the measurement estimator is shown in Fig. 2. The inputs to the network are the measurements from a study system and the output units provide the inputs' estimated values. Thus, the number of input and output units are the same and equal to the number of measurement variables. The number of units in the hidden layer usually depends on the number of input units, the size of training examples and the system being studied. A reasonable size for the hidden layer can be determined by a trial-and-error procedure. Our experience indicates that sufficient accuracy can be obtained by using one hidden layer and an identical number of units in the input, hidden, and output layers.

Given that the true value of measurement variables could be either positive or negative, the activation function we have developed for the units in the hidden and output layers is given by

$$o_i = \frac{1 - \exp(-\text{net}_i)}{1 + \exp(-\text{net}_i)} \quad (1)$$

where net_i and o_i are the input and output, respectively, of the ith unit. This is a smooth version of a $(-1, 1)$ step function. When the generalized delta rule is used as a training procedure, the error signal of the ith unit in the output layer is given by

$$\delta_i = \tfrac{1}{2}(1 - o_i^2)(t_i - o_i) \quad (2)$$

where t_i is the desired output of the ith unit. The error signal of the ith unit in the hidden layer is given by

$$\delta_i = \tfrac{1}{2}(1 - o_i^2)\sum_k \delta_k w_{ki} \quad (3)$$

where k is the unit index in the output layer and w_{ki} is the weight from the ith to kth unit.

2.2. Training examples

Neural networks learn exactly what they are asked to learn. In other words, the training examples must provide the network with all information they need to construct a desired input–output mapping. Thus, training examples play a very important role in the success of neural networks. In the case of bad-data detection and identification problems, training examples have to cover various typical load and generation patterns, and several feasible and most probable changes in the system topology within the entire demand space. For this reason, load changes of load buses, generation schedules of generating units, and tap settings of transformers must be investigated.

Although the possible system operating states seem to be infinite, training a network needs only a reasonable number of training examples consisting of typical system operating states. This is mainly because a trained neural network is a good interpolator. To construct a reasonably complete set of training examples, system operating experience is strongly required. Also, the typical system operating conditions chosen for training purposes must satisfy the requirements of load forecasts, optimal dispatch policies, and system operational planning such as unit commitment, maintenance, etc. Therefore, the characteristics of the system must be known very well. To obtain a set of training examples for a given system, in our method the demand of each load bus is represented by a load range model which uses a band of load values to cover all possible load changes. A set of typical load combinations can then be formed by choosing a number of data points from each load range model. An example of a load range model is presented in Section 3 of the paper. The use of a load range model greatly reduces the actual size of training examples. For example, the maximum demand at some buses may never coincide with the minimum demand at some other buses in an actual power system. Thus, the simultaneous occurrence of the maximum and minimum demands at various system buses may never appear in the training examples. The corresponding operating states of generating units and transformers are determined according to the load distribution. After all typical operating conditions are determined, the true value of measurement variables under these conditions is computed by a number of standard power flow studies. The computed values of measurement variables are the input–desired-output pairs of the training patterns. Finally, the training patterns are normalized to within a $(-1, 1)$ range to match the activation function given in Eq. (1).

Topology changes of the system include the outage of transmission lines, generating units and transformers. The normal topology changes in a real power system are limited by several constraints such as load demand, generation setting, system stability and security, economic operation and energy interchange with other utilities, etc. However, there will still be many different topology changes in a large power system. Since the case in each topology is different, the use of a single training set for all different topologies will force the weights to learn an average function mapping. Therefore, one needs to construct a training set for each individual system topology. Each individual training set will have its own trained connection weights when they are used to train a chosen neural network. A series of different trained connection weights can then be selected online to match various system topology changes.

In a large-scale power system, the number of measurement variables may become quite large. One possible way of applying the proposed method to large power systems is through the use of a technique which we call the measurement divide technique. This is just a data processing technique. Using this measurement divide technique, the large number of measurement variables is divided into several subsets with reasonable sizes suitable for present computing systems. Each measurement subset has its own neural network measurement estimator. In this approach note that the power system itself is not divided. At the training stage, the true values of measurement variables in all training examples are obtained from the power flow studies conducted over the entire power system. The resultant measurements are then divided into several measurement subsets each with an assigned neural network, as shown in Fig. 3. Many training patterns for each neural network are formed by selecting the true values of its corresponding measurement subset from the training examples. Each training pattern represents a training example for its corresponding neural network. Each neural network is actually trained to be an encoder, encoding its measurement subset onto itself. Such a trained network is called autoassociative. After train-

ing, each trained network becomes an estimator of a corresponding measurement subset. Since all neural networks are trained by using the measurement subsets derived from the same training examples, these neural networks will work together as a complete measurement estimator for the whole power system. In an online application, the raw measurements received by the preestimation filter are first divided into several measurement subsets, the same as those at the training stage of the procedure. These subsets are then sent to their corresponding trained networks. Each neural network estimates the values of its own measurement subset, and then sends them to the bad-data detection and identification block of the filter. In this block, the difference between the measured and estimated values of each measurement variable is computed and is then compared with a given identification threshold to flag bad measurements.

There are some advantages in using the proposed measurement divide technique. This technique reduces the time needed to train a large neural network and it increases the online speed of the neural network model. For example, if a measurement system has 100 variables and uses a single neural network, there will be $100 \times 100 + 100 \times 100 = 20\,000$ weights. However, if the measurement variables are divided into four subsets with equal sizes, each neural network will have only $25 \times 25 + 25 \times 25 = 1250$ weights. Moreover, the parallel processing nature of these neural networks provides a higher speed for online detection and identification of bad data. The proposed measurement divide technique will also reduce the number of topology changes required to train each neural network. For a subset of measurement variables, the effects of some topology changes outside this subset will be very small. Thus, these 'external' topology changes can be ignored when each neural network is being trained. A large number of measurements, however, should not be divided into very small subsets, because a small neural network cannot be trained by a large training set and it cannot deal with multiple bad data satisfactorily. An example is given in Section 3 of the paper to show exactly how the measurement divide technique works.

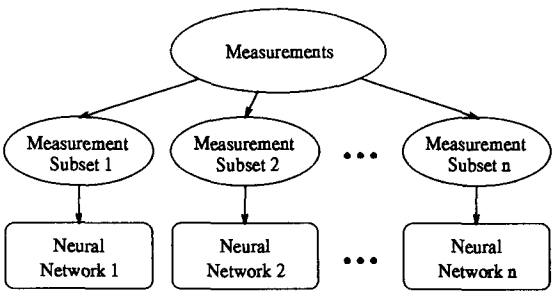

Fig. 3. Neural network measurement estimator model for large power systems.

2.3. Detection and identification method

At the training stage, the connection weights of the network are corrected with a training error to acquire a relationship among its measurement variables. The training patterns and the values of their parameters are all related to each other as they conform with the system power flow equations. Thus, the proposed model is easy to train and the training errors are smaller than a standard deviation of the measurements when the best possible weights are achieved. When a trained neural network receives a good raw measurement set, which it may or may not have seen before, the estimated values of the measurement variables, i.e. the outputs of the neural network, are computed and will be very close to the input measurements. This is indicated by a very small difference between the measured and estimated values of each measurement variable. On the other hand, if the trained network is presented with measurements containing some bad data, the measurement errors are distributed among all hidden units and then among all output units through the distributed associated weights. Since the activation value of each unit in the network depends on all inputs, the effects of bad data on the outputs of the neural network are shared by all measurement variables. Consequently, the estimated value of each measurement variable will contain a reasonable level of error. Although the estimated errors for bad data are bigger than those for the good ones, they are still close to their true values, that is to say, they are far away from their original measured values.

In the bad-data detection and identification block of the filter, comparison between measurement inputs and their corresponding estimated values will flag the bad measurement inputs. The flag is raised when the difference between a bad measurement and its estimated value is much larger than that of a good measurement value. The rule for bad-data detection and identification is as follows:

$$(z_i - o_i)^2 > r_i^2 \qquad i = 1, \ldots, n \tag{4}$$

where z_i is the measured value of the ith measurement variable, o_i the estimated value of the ith measurement variable, and r_i the threshold value of bad data for the ith measurement.

Using this rule, if the square of the difference between the measured and estimated values of a measurement variable is greater than a given threshold, this value is flagged as a bad measurement. The threshold of each variable is determined to be larger than its measurement standard deviation σ_i and less than a critical error used in defining the bad data. In our case studies, we have used a threshold $r_i = 10\sigma_i$ where i is the measurement index. Depending on the desired accuracy, other threshold values may also be chosen.

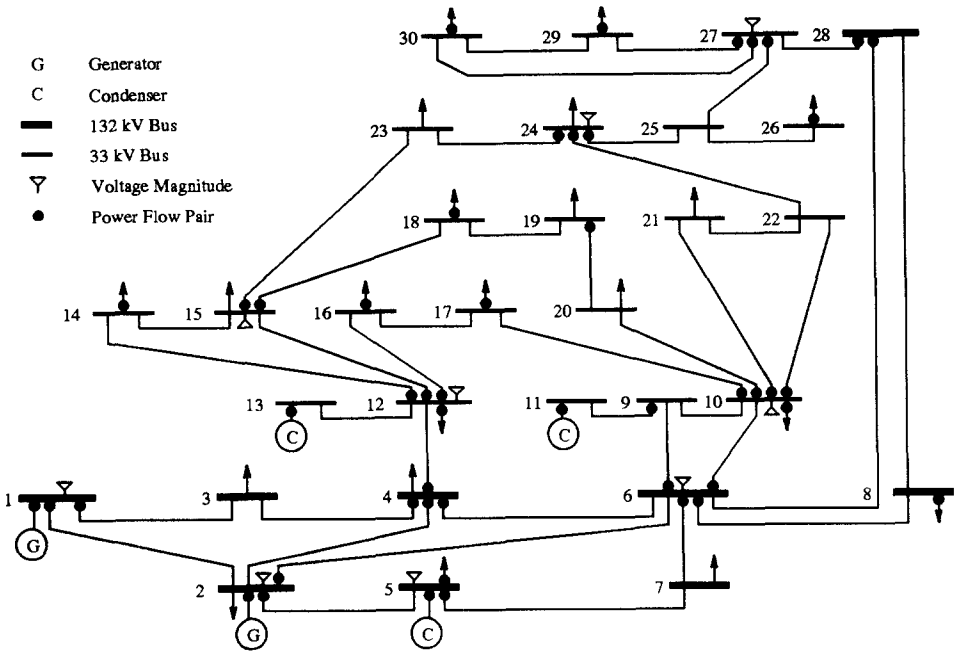

Fig. 4. IEEE 30-bus system.

2.4. Replacement of bad data

Unlike the traditional state estimation methods, the proposed filter can detect, identify, and provide good estimates of most bad measurements simultaneously. If there are too many bad measurements in the set, then their estimated values may not be very accurate. However, these not-so-accurate estimates can still replace the corresponding bad measurements to obtain better estimates in another run of the proposed detection and identification process. This iteration process may need to be repeated at most two or three times to obtain reasonably accurate estimates of the bad measurement data. In the example of Section 3, case 2, there are 103 measurements where 10 of them are bad. The errors of bad data are $\pm(20-100)$ times their measurement standard deviations. After two or three quick iterations, the estimated errors of bad measurements are close to their standard deviations.

For economic and cost saving purposes, a measurement system might be designed with minimum redundancy. The elimination of identified bad data could then cause this measurement system to lose its local or global redundancy. Thus, the system might become unobservable. Using the proposed filter, however, if an identified bad measurement happens to be a critical measurement, it could be replaced with a reasonably accurate estimated value in order to maintain the system observability. Obviously, this estimated value is better than replacing bad data by crude pseudomeasurements. Using such a bad-data replacement scheme, small reliable measurement systems could be designed with a minimum investment. The results of a study using this scheme are provided in the next section.

3. Study results

The proposed preestimation filter has been tested using the IEEE 30-bus system with the measurement configuration shown in Fig. 4. For a better presentation of the results, our discussion focuses on the basic topology of this test system. Various topology changes may be studied.

In the absence of actual data, the system operating conditions are assumed to be as follows. The load demands given in the original 30-bus system are assumed to be the maximum load demands. The minimum loads are assumed to be 20% of the maximum values. Load range models are used to represent the load changes at load buses. An example is shown in Fig. 5, where the maximum and minimum load values at a load bus are simply defined by a six-step line. All possible load changes at this load bus are covered by the shaded area in between the maximum and minimum load values. The same logic is used for the

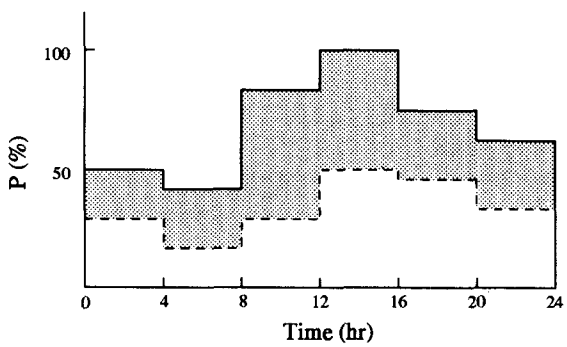

Fig. 5. Load range model.

Table 1
Identification results of single bad data

Bad measurement	Standard deviation	True value t_i	Measured value z_i	Estimated value o_i	Measured error $z_i - t_i$	Estimated error $o_i - t_i$	Variation $\|z_i - o_i\|$
P_{6-9}	0.010	0.1878	−0.4583	0.1863	−0.6464	−0.0015	0.6446
P_{18}	0.002	−0.0199	−0.1353	−0.0171	−0.1154	0.0028	0.1183
V_{15}	0.005	1.0248	1.3635	1.0289	0.3387	0.0041	0.3346
Q_{6-8}	0.010	0.0980	0.3480	0.0954	0.2500	−0.0026	0.2526
P_{10-20}	0.005	0.0780	0.4495	0.0735	0.3715	−0.0045	0.3760

generation at bus 2 of the system. At bus 2, the maximum generation is assumed to be 0.4 p.u. and the minimum generation is 20% of this value. Bus 1 is taken as the reference bus. Tap settings of the four transformers are within a 0.95–1.05 range.

Using the load range model, 1500 different load combinations together with their corresponding generation schedules and transformer tap settings are selected to form the typical system operating states. The system load flows are then computed to determine the true values of the measurement variables for building the training patterns. The measurement system consists of 103 measurement variables containing nine voltage magnitudes, 16 real and reactive power injection pairs, and 31 real and reactive power flow pairs. Depending on the maximum values of the measurement meters, the standard deviation of all power measurements is assumed to be 0.01–0.001 p.u.; the standard deviation of voltage magnitudes is assumed to be 0.005 p.u. Using the measurement divide technique discussed in Section 2.2, the measurements are divided into two subsets, one for the 132 kV transmission subsystem and another for the 33 kV distribution subsystem. The 132 kV subset includes 48 measurements at buses 1–9, 11, 13, and 28. The 33 kV subset contains 55 measurements at 18 other buses. The frameworks of the neural networks used for the two subsets are $48 \times 48 \times 48$ (i.e. 48 input units, 48 hidden units, and 48 output units) and $55 \times 55 \times 55$, respectively.

The network training is performed with an adaptive learning rate on an IBM RISC/6000 workstation. Whenever the connection weights of the network are updated, the direction of the error gradient is also determined. The correlation between the direction of successive error gradients is then computed and used to modify the learning rate and the momentum of the generalized delta rule. The initial learning rate and the momentum are set equal to 0.1. The maximum values of the learning rate and momentum which appeared during the training procedure are about 0.5 and 0.9, respectively. Four cases are studied to determine the ability of the preestimation filter to identify single and multiple interactive and conforming bad data, and to replace bad data with their best estimates.

3.1. Case 1. Single bad data

In this case, three hundred system operating states are created by arbitrarily choosing the load demand from a number of load range models of load buses and their corresponding generations and transformer tap settings under assumed system operating conditions. Single bad data are then randomly introduced into the measurements of each operating state. The errors of bad data are ±(20–100) times their measurement standard deviations. The identification thresholds r_i in Eq. (4) are set equal to 10 times their measurement standard deviations. For each operating state, the preestimation filter accurately identified the bad data. The average estimated error of the bad data after the third iteration of our detection and identification procedure is about 0.01 p.u. The identification results of the first five operating states are listed in Table 1.

3.2. Case 2. Multiple bad data

The same three hundred system operating states studied in case 1 are used, except that now ten bad data are randomly generated in the measurements of each operating state. The results using the same identification threshold as in case 1 show that the filter identified all of the bad data. The filter also rejected a few good measurements as bad data due to the presence of several very large bad measurements in some operating states. The correct identification rate (the number of identified real bad data divided by the number of identified bad data) turns out to be 97.5%, and the average estimated error of the bad data in the third iteration is 0.018 p.u. The results are reported in Table 2.

3.3. Case 3. Conforming bad data

Here is an example of identifying conforming bad data. Six conforming bad data are presented at bus 1 by changing the sign of the active and reactive power injections P_1 and Q_1, and the active and reactive power flows P_{1-2}, Q_{1-2}, P_{1-3}, and Q_{1-3} in one of the system operating states. Using the conventional largest normalized residual test in the WLS estimator, none of the conforming bad data were identified. However, the

Table 2
Identification results of ten bad data

Bad measurement	Standard deviation	True value t_i	Measured value z_i	Estimated value o_i	Measured error $z_i - t_i$	Estimated error $o_i - t_i$	Variation $\|z_i - o_i\|$
P_{6-8}	0.010	0.2389	1.0451	0.2486	0.8062	0.0097	0.7965
Q_{6-10}	0.002	−0.0295	0.1427	−0.0273	0.1722	0.0022	0.1700
P_{6-9}	0.010	0.2528	−0.5603	0.2420	−0.8131	−0.0108	0.8023
P_{4-12}	0.010	0.3763	−0.0823	0.3645	−0.4586	−0.0118	0.4468
Q_{13}	0.010	0.2364	−0.3913	0.2194	−0.6277	−0.0170	0.6107
P_{15-18}	0.003	0.0510	−0.0934	0.0430	−0.1444	−0.0080	0.1364
P_{29}	0.001	−0.0219	0.0218	−0.0193	0.0437	0.0026	0.0411
P_{17}	0.005	−0.0823	−0.2008	−0.0696	−0.1185	0.0127	0.1312
P_{10-21}	0.008	0.1402	0.3277	0.1264	0.1875	−0.0138	0.2013
Q_{12-14}	0.001	0.0186	0.0437	0.0167	0.0251	−0.0019	0.0270

WLS estimator identified three good measurements P_2, Q_{4-3}, and $P_{4\ 3}$ as bad data. After eliminating these three 'bad data', the largest normalized residual is 1.01 (assumed threshold is 3), and a very poor state estimation result is produced. The proposed filter, however, correctly identified all of the actual bad data. After eliminating the six conforming bad data, the system is still observable. The state estimates obtained by using the remaining measurements are very close to the standard power flow solutions. Table 3 shows the WLS largest normalized residuals along with the cost function values. Table 4 shows the results from the proposed preestimation filter.

3.4. Case 4. Bad-data replacement

To maintain the system observability, bad-data replacement is performed in this case. Six bad data are introduced with random gross errors. They are two power injection pairs at buses 12 and 14, and one power flow pair P_{12-14}, Q_{12-14}. The elimination of bad measurements P_{12}, Q_{12}, P_{14}, and Q_{14} makes the P_{12-14} and Q_{12-14} measurements critical. Thus, if all of these six bad measurements are eliminated, the system becomes unobservable. The proposed preestimation filter identified all of these bad data, and the results are given in Table 5. Here, four of the six identified bad data are eliminated, and the critical measurements P_{12-14} and Q_{12-14} are replaced by their estimated values in order to maintain the system observability. The resulting measurements are then sent to a WLS estimator where, except for the estimates at bus 14, the results are exactly the same as those when perfect measurements are used. The estimated voltage and phase angle at bus 14 are 1.0258 p.u. and −14.06° when perfect measurements are used, and 1.0251 p.u. and −14.10° when modified measurements are used.

Table 3
Identification results of conforming bad data using the WLS largest normalized residual test

Iteration	$J(x)$	1st largest normalized residual		2nd largest normalized residual		3rd largest normalized residual	
1	2982.68	P_2	−38.99	P_{1-2}	−27.89	P_{1-3}	−27.74
2	1629.13	Q_{4-3}	−39.20	Q_{1-3}	38.24	P_{4-3}	−22.79
3	112.14	P_{4-3}	10.53	P_{1-3}	−9.99	P_{1-2}	4.48
4	1.45	V_1	1.01	Q_2	−0.59	P_{1-2}	−0.39

Table 4
Identification results of conforming bad data using the proposed neural network model

Bad measurement	Standard deviation	True value t_i	Measured value z_i	Estimated value o_i	Measured error $z_i - t_i$	Estimated error $o_i - t_i$	Variation $\|z_i - o_i\|$
P_{1-2}	0.010	1.5228	−1.5228	1.4968	−3.0456	−0.0260	3.0196
P_{1-3}	0.010	0.7244	−0.7244	0.7116	−1.4487	−0.0128	1.4360
P_1	0.010	2.2471	−2.2471	2.1983	−4.4942	−0.0488	4.4454
Q_1	0.010	−0.1858	0.1858	−0.1787	0.3716	0.0071	0.3645
Q_{1-2}	0.010	−0.1968	0.1968	−0.1885	0.3936	0.0083	0.3853
Q_{1-3}	0.001	0.0111	−0.0111	0.0108	−0.0222	−0.0003	0.0219

Table 5
Identification results in bad-data replacement case

Bad measurement	Standard deviation	True value t_i	Measured value z_i	Estimated value o_i	Measured error $z_i - t_i$	Estimated error $o_i - t_i$	Variation $\|z_i - o_i\|$
P_{14}	0.005	−0.0470	0.1555	−0.0418	0.2025	0.0052	0.1973
P_{12}	0.008	−0.1101	0.3362	−0.0923	0.4463	0.0178	0.4285
Q_{12}	0.005	−0.0718	−0.1956	−0.0843	−0.1238	−0.0124	0.1113
Q_{14}	0.001	−0.0156	0.0284	−0.0135	0.0440	0.0021	0.0419
P_{12-14}	0.005	0.0625	0.2209	0.0657	0.1584	0.0032	0.1552
Q_{12-14}	0.001	0.0205	0.0524	0.0218	0.0319	0.0013	0.0306

4. Conclusions

A preestimation bad-data detection and identification filter based on back-propagation neural networks is explored. Due to the parallel distributed processing nature of the neural networks, the preestimation filter can quickly identify single and multiple conforming and nonconforming bad measurements. The filter can replace bad measurements with their corresponding estimated values to build a reliable database in a control center. Once trained offline, the filter may be used online to identify bad data before estimation and thus improve the efficiency of the conventional state estimators. The results of the proposed approach indicate that the accuracy of bad-data identification depends on the training examples. Designing a complete set of training examples, choosing the right size of the hidden layer, and actually training a network could be very tedious and expensive. Trained neural networks are good interpolators but not good extrapolators. Thus, there is a possibility of an incorrect response when a network is exposed to a new situation for which it has not been trained. However, the proposed filter is a preestimation tool for bad-data processing. Bad data which the filter fails to identify could then be identified by the standard postestimation analysis.

References

[1] F.C. Schweppe, J. Wildes and D.B. Rom, Power system static state estimation, Parts I, II, III, *IEEE Trans. Power Appar. Syst., PAS-89* (1970) 120–135.

[2] R.E. Larson, W.F. Tinney, L.P. Hadju and D.S. Piercy, State estimation in power systems, Parts I and II. Theory and feasibility, *IEEE Trans. Power Appar. Syst., PAS-89* (1970) 345–362.

[3] H.M. Merrill and F.C. Schweppe, Bad data suppression in power system static state estimation, *IEEE Trans. Power Appar. Syst., PAS-90* (1971) 2718–2725.

[4] E. Handschin, F.C. Schweppe, J. Kohlas and A. Fiechter, Bad data analysis for power system state estimation, *IEEE Trans. Power Appar. Syst., PAS-94* (1975) 329–337.

[5] A. Garcia, A. Monticelli and P. Abreu, Fast decoupled state estimation and bad data processing, *IEEE Trans. Power Appar. Syst., PAS-98* (1979) 1645–1652.

[6] W.W. Kotiuga and M. Vidyasagar, Bad data rejection properties of weighted least absolute value techniques applied to static state estimation, *IEEE Trans. Power Appar. Syst., PAS-101* (1982) 844–851.

[7] Xiang Nian-de, Wang Shi-ying and Yu Er-keng, A new approach for detection and identification of multiple bad data in power system state estimation, *IEEE Trans. Power Appar. Syst., PAS-101* (1982) 454–461.

[8] L. Mili, Th. Van Cutsem and M. Ribbens-Pavella, Hypothesis testing identification: a new method for bad data analysis in power system state estimation, *IEEE Trans. Power Appar. Syst., PAS-103* (1984) 3239–3252.

[9] F. Zhuang and R. Balasubramanian, Bad data processing in power system state estimation by direct data deletion and hypothesis tests, *IEEE Trans. Power Syst., PWRS-2* (2) (1987) 321–330.

[10] I.W. Slutsker, Bad data identification in power system state estimation based on measurement compensation and linear residual calculation, *IEEE Trans. Power Syst., 4* (1) (1989) 53–60.

[11] B.M. Zhang and K.L. Lo, A recursive measurement error estimation identification method for bad data analysis in power system state estimation, *IEEE Trans. Power Syst.*, 6 (1) (1991) 191–198.

[12] G.N. Korres and G.C. Contaxis, A reduced model for bad data processing in state estimation, *IEEE Trans. Power Syst., 6* (2) (1991) 550–556.

[13] D.C. Park, M.A. El-Sharkawi, R.J. Marks II, L.E. Atlas and M.J. Damborg, Electric load forecasting using an artificial neural network, *IEEE Trans. Power Syst., 6* (2) (1991) 442–449.

[14] D.J. Sobajic and Y.H. Pao, Artificial neural-net based dynamic security assessment for electrical power systems, *IEEE Trans. Power Syst., 4* (1) (1989) 220–228.

[15] S. Weerasooriya, M.A. El-Sharkawi, M. Damborg and R.J. Marks II, Towards static-security assessment of a large-scale power system using neural networks, *IEE Proc. C, 139* (1992) 64–70.

[16] R.K. Hartana and G.G. Richards, Harmonic source monitoring and identification using neural networks, *IEEE Trans. Power Syst., 5* (4) (1990) 1098–1104.

[17] Z. Ouyang and S.M. Shahidehpour, A hybrid artificial neural network–dynamic programming approach to unit commitment, *IEEE Trans. Power Syst., 7* (1) (1992) 236–242.

[18] D.E. Rumelhart, J.L. McClelland and the PDP Research Group, *Parallel Distribution Processing: Explorations in the Microstructure of Cognition*, Vol. 1, MIT Press, Cambridge, MA, 1986.