Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007)

Combined Training of Recurrent Neural Networks with Particle Swarm Optimization and Backpropagation Algorithms for Impedance Identification

Peng Xiao, Ganesh K. Venayagamoorthy, and Keith A. Corzine
Real-Time Power and Intelligent Systems Laboratory
Department of Electrical and Computer Engineering
University of Missouri-Rolla, USA
pxfx7@umr.edu, gkumar@ieee.org, and keith@corzine.net

***Abstract* – A recurrent neural network (RNN) trained with a combination of particle swarm optimization (PSO) and backpropagation (BP) algorithms is proposed in this paper. The network is used as a dynamic system modeling tool to identify the frequency-dependent impedances of power electronic systems such as rectifiers, inverters, and DC-DC converters. As a category of supervised learning methods, the various backpropagation training algorithms developed for recurrent neural networks use gradient descent information to guide their search for optimal weights solutions that minimize the output errors. While they prove to be very robust and effective in training many types of network structures, they suffer from some serious drawbacks such as slow convergence and being trapped at local minima. In this paper, a modified particle swarm optimization technique is used in combination with the backpropagation algorithm to traverse in a much larger search space for the optimal solution. The combined method preserves the advantages of both techniques and avoids their drawbacks. The method is implemented to train a RNN that successfully identifies the impedance characteristics of a three-phase inverter system. The performance of the proposed method is compared to those of both BP and PSO when used separately to solve the problem, demonstrating its superiority.**

I. INTRODUCTION

There has been plenty amount of research on dynamic system modeling with recurrent neural networks [1-4]. RNNs have internal feedback loops within the network which allows them to store previously presented patterns. The capability makes this type of neural networks superior to conventional feedforward neural networks in modeling dynamic systems because the network outputs are functions of both the current inputs as well as their internal states.

In this paper RNN is used to extract the frequency-dependent impedance characteristics of power electronics systems. Traditional impedance measurement techniques [5, 6] require multiple injections of test signals of different frequencies into the system during its normal operation. The test procedure is often complex and prolonged, which is not efficient and convenient, especially when the system under test is critical and minimal interruption is desired. The impedance identification method proposed in this paper avoids multiple injections and only one set of data is required; thus greatly reduce disturbances to system operation.

An important step of the RNN-based impedance extraction method is to train the neural network such that it learns the dynamic behavior of the tested system. There are a variety of training algorithms available for neural networks [7-11]. However, certain properties of the RNN make many of the algorithms less efficient, and it often takes an enormous amount of time to train a network of even a moderate size. In addition, the complex error surface of the RNN network makes many training algorithms more prone to being trapped in local minima.

A recent development in evolutionary computation techniques has enabled the application of various population-based search algorithms in the training of neural networks. Many researchers have focused on using genetic algorithms to tune the weight parameters of various neural networks [7, 8].

As a relatively new stochastic algorithm, the particle swarm optimization method has gained more and more attention. PSO is inspired by social interaction of knowledge, where each particle in a swarm flies through the search space with a velocity that is dynamically adjusted according to its own and its companion's historical behaviors. The PSO algorithm is easy to implement and has been empirically shown to perform well on many optimization problems.

Like other evolutionary techniques, PSO was quickly used in neural network training. A preliminary study of PSO trained feedforward networks was presented in [9]. Test results based on the training of some simple problems showed that the performance of PSO is not much better than other methods. However, the authors argued that PSO is still promising in cases where a high number of local minima are known to exist. Another research of PSO trained neural network showed its superior learning ability for the XOR and simple recurrent network examples [10]. However, the authors also pointed out that PSO performed poorly in the learning of natural language phrase parsing. A detailed performance comparison of PSO and backpropagation algorithms was presented in [11], which demonstrated that PSO needs less computational resources to achieve the same error goal as with BP. However, only feedforward network was considered and the example problem used was straightforward.

This paper investigates the performance of PSO and BP algorithms in the training of recurrent neural networks. Based on extensive test results, it shows the advantages and weaknesses of both methods. In addition, a training procedure that combines both techniques is proposed to avoid their shortcomings while preserving their advantages. The proposed training method is applied in impedance identification problem of power electronic systems.

The rest of this manuscript is organized as follows. Problem formulation is presented in Section II, where the impedance identification issue is described. Section III discusses how to use RNN to extract the impedance information. In section IV,

1-4244-0708-7/07/$20.00 ©2007 IEEE

the training of RNN is described in detail, where the performances of BP and PSO are compared. The combined training technique is also presented. Section V demonstrates the superior performance of the proposed method with simulation results.

II. IMPEDANCE IDENTIFICATION OF POWER ELECTRONIC SYSTEMS

Due to the wide-spread application of power electronic devices in modern power distribution systems, an important issue that attracts more and more research work is the stability of the systems where multiple power electronic devices interact with each other. Most power electronic switching devices have unique dynamic characteristics and stability problems that are not well understood due to the nonlinearity and time dependency of converters. Their constant-power operation capability often leads to negative impedance characteristic, which is a common cause for system instability [12].

Over the past decade various stability criteria have been established in terms of source and load impedances or admittances for both dc and ac systems [13-15]. To utilize these criteria for stability analysis, it is often necessary to obtain the frequency-dependent impedances/admittances of a subsystem by experiment. A widely used impedance measurement method is through the injection of perturbation signals. Figures 1 and 2 illustrate the shunt injection diagrams for both dc and three-phase systems where a small current signal of a certain frequency f_i is injected into the system source/load interface. During the injection, the interface voltage and the source and load currents are measured and recorded. For a dc interface, the small-signal source impedance and load admittance at frequency f_i can be determined by

$$Z_s(f_i) = -\frac{V(f_i)}{I_s(f_i)} \qquad Y_l(f_i) = \frac{I_l(f_i)}{V(f_i)} \qquad (1)$$

where all the variables are complex numbers. Similar equations can be derived for ac systems, although a reference frame transformation is often involved [5].

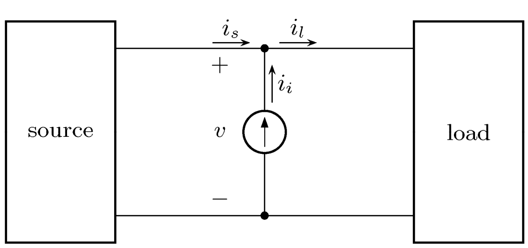

Fig. 1. Impedance Measurement in dc/single-phase system.

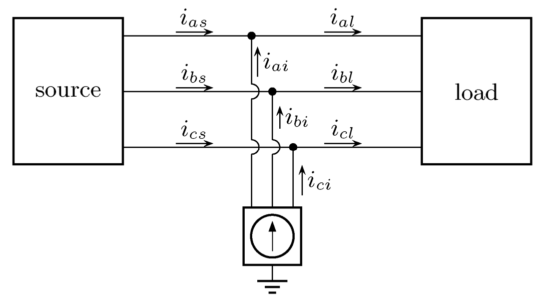

Fig. 2. Impedance Measurement in three-phase system.

To obtain accurate characteristics of a system over a wide frequency range, multiple injections of voltage or current perturbation signals of different frequencies are often required. The main disadvantages of this impedance measurement procedure include:

1. It takes a considerable amount of time to complete the injections for all the frequencies and take the measurements.
2. The operating point of the system may vary during the prolonged test procedure, which can lead to inconsistency in the measured system impedance characteristics.
3. The injection device often consumes a significant amount of power and heat dissipation can be a serious issue.
4. The procedure only gives impedance information at the injection frequencies.

To solve these problems, the key point is to minimize the injection time. In this paper, the recurrent neural network is used to model the system and extract the impedance information.

III. IMPEDANCE IDENTIFICATION WITH RNN

Unlike the widely used multi-layer perceptron network (MLPN) where data flows only in the forward direction, recurrent neural networks are feedback networks in which the present activation state is a function of the previous activation state as well as the present inputs. The feedback mechanism provides a memory to the recurrent networks so that they are capable of modeling systems with internal dynamics.

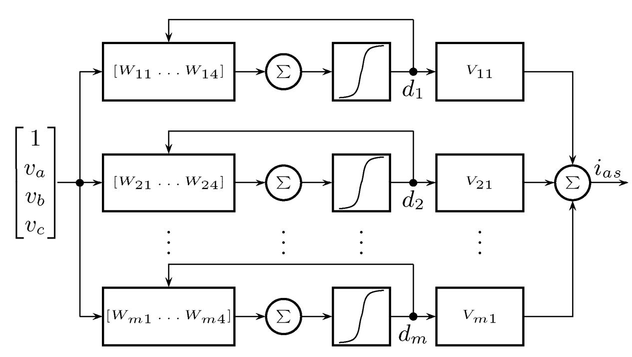

Fig. 3. Topology of the Elman recurrent network.

The block diagram of a two-layer Elman RNN with three voltages as input and one current as output is shown in Fig. 3, where *m* neurons are used in the hidden layer. The topology is similar to that of a feedforward network, except that the outputs of the hidden layer is used as the feedback signals. Although not shown in the diagram, there is a one-step time delay in the feedback path so that previous outputs of the hidden layer, also called the states of the network, are used to calculate new output values. For a network with *l* inputs, *m* hidden neurons, and *n* outputs, the hidden layer equations are

$$s_k(t) = \sum_{i=1}^{l} w_{ik}^{(1)} x_i(t) + \sum_{j=1}^{m} w_{jk}^{(2)} d_j(t-1) \tag{2}$$

$$d_k(t) = \mathrm{sgm}(s_k(k)) \tag{3}$$

where $\boldsymbol{x}(t)$ is the input vector, $\boldsymbol{w}^{(1)}$ is the weight matrix associated with the inputs and hidden neurons, and $\boldsymbol{w}^{(2)}$ is the weight matrix associated with the states and hidden neurons. The hyperbolic tangent sigmoid function is selected as the nonlinear activation function sgm(·) of the hidden layer because it can provide a dual polarity signal to the output.

Linear neurons are used in the network's output layer. The outputs are determined by

$$y_k(t) = \sum_{i=1}^{m} w_{ik}^{(3)} d_i(t) \tag{4}$$

where $\boldsymbol{w}^{(3)}$ is the weight matrix associated with the hidden neurons and the outputs.

Past research has demonstrated the ability of the RNN to learn process dynamics and provide efficient forecasts, and it has found application in many areas such as wind speed and power forecasting [1], design of a power system stabilizer [2], induction motor speed estimation[3], and prediction of elephant migration [4].

To identify the impedance characteristics of a system with RNN, it is necessary to inject signals to the system and measure currents and voltages. However, instead of multiple injections of signals of different frequencies, a sampled uniform random signal is injected to perturb the system and only one set of voltage and current measurements is needed. The measured signals are used as inputs and target outputs to train a RNN. A well trained RNN then has the ability to approximate the time-domain dynamic responses of the system to different input voltages. To extract the frequency-domain information, simulated voltage signals are fed into the network to obtain its current output. These signals can then be used in (1) to determine the impedance and admittance of the system.

A significant advantage of the RNN based impedance identification method is that only one perturbation injection is needed, thus the amount of time for conducting tests on the system is greatly reduced. The majority of the identification work is done offline with the measured signals. In addition, once the training process is finished, ideal injection signals can be applied to the trained network to determine the impedance characteristics, which makes the results less susceptible to noises and frequency leakage.

IV. RNN TRAINING WITH BP AND PSO

The effectiveness of a RNN to accurately extract the impedance characteristics of a system depends on several factors. Firstly, the structure network must be designed so that it is able to model the tested system as close as possible. The important parameter here is the number of hidden neurons, which is the same as the number of states of the network. Generally more hidden neurons would enable the network to simulate more complex and higher-order dynamic systems. Secondly, the perturbation signals must have a wide spectrum that covers the frequency range of interest. They also should have a magnitude that is high enough to counter the effects of measurement noise. Finally, given a well-designed RNN network and effective injection signals, the network needs to be trained so that it can mimic closely the dynamic behaviors of the tested system. This section deals with the RNN training issue with several different approaches.

A. Backpropagation training algorithm

The backpropagation algorithm is by far the most commonly used training method for static multi-layer feedforward neural networks. The standard backpropagation is a gradient descent algorithm, in which the network weights are moved along the negative of the gradient of the performance function. The gradients are determined by performing computations backward through the network. Although the algorithm and many of its variants suffer from some drawbacks such as slow convergence and being trapped in local minima, it has been shown to be effective to train feedforward neural networks in many applications.

The same gradient-based backpropagation algorithm can be used to train recurrent neural networks. However, because of the existence of feedback loops in the network structure, the computation of gradients becomes much more complex, which makes the backpropagation procedure computationally more intensive. In addition, the error surfaces for recurrent networks are more complex than those for static networks, therefore the training is more likely to be trapped in local minima.

To help the BP algorithm get out of a local minima area, one common strategy is to restart the training when the gradient is below a very low threshold while the output error is still large. By restarting the training process, a different set of random initial weights are selected, which may hopefully lead to an optimal solution. However, several properties of the RNN makes this technique ineffective. Firstly, with the feedback loops RNN often has a much larger set of weight parameters to determine. This means that the search space of RNN is of a much higher dimension than that of feedforward networks. The random initialization of weights in the huge search space is less likely to fall near the desired solution. Secondly, as mentioned above, the BP algorithm requires much more computational efforts in computing the output and backpropagation. Thus discarding previous training results and restarting the whole process is very inefficient.

In the past, considerable efforts have been devoted to optimally initialize the weight parameters of a neural network before its training begins. A multidimensional geometrical approach was proposed to accelerate the training speed of feedforward networks [16]. The underlying idea is to initialize the parameters of the network so that all activation functions can always be operated in an active region. Some researchers suggested that a simulated annealing method or a genetic algorithm should be used for initializing the weights between the input and the hidden layers [17]. Much of the focus of research has been on preventing the training from starting in the saturated region.

B. PSO training algorithm

Particle swarm optimization is a form of evolutionary computation algorithm based on the social metaphor of bird flocking or fish schooling. Like the genetic algorithm (GA), PSO is a population (swarm) based optimization tool. It has constructive cooperation between particles, and particles in the swarm can share information. PSO has memory of the past, so knowledge of good solutions is retained by all particles.

The application of PSO in neural network training involves creating a swarm of networks initialized with random weights. Each network is called a particle and is a candidate solution. The particles have the ability to retain their own best-ever state and communicate with each other. The swarm evolves in the search space by letting all the particles fly towards the best solutions they know of. In the local version of the PSO algorithm, each particle only communicates its neighbors; while in the global version, each particle can communicate with any other particles so everyone knows where the best-solution-so-far is.

The whole training process with the global version PSO can be summarized as follows.

1. Initialize the weight parameters of N networks (particles) with random numbers, where N is the number of particles in the swarm. Also initialize N velocity vectors $v(0)$ with random numbers.
2. Start the iteration by feeding each network with the training data, and calculating its mean squared error (MSE), which is used as the fitness criterion of the particles.
3. Compare the MSE of each particle with its best history value, also called the personal best (pbest) MSE. If the current MSE is lower than the pbest MSE, update pbest MSE and store current weights as the pbest weights.
4. Find the minimal newly calculated MSE in the swarm.
5. Compare the minimal MSE with the global best (gbest) MSE. If the minimal MSE is lower than gbest MSE, update gbest MSE and store the corresponding weights as the gbest weights.
6. Update the velocity vector of each particle with

$$\begin{aligned} v_i(t+1) = wv(t) &+ c_1\rho_1\left(W_{pbest} - W_i\right) \\ &+ c_2\rho_2\left(W_{gbest} - W_i\right) \end{aligned} \tag{5}$$

where w is the inertia term, c_1 and c_2 are acceleration terms, ρ_1 and ρ_2 are uniformly distributed random numbers in [0, 1], and W_{pbest}, W_{gbest}, and W_i are the weight vectors of pbest, gbest, and the current particle, respectively.

The iteration loop continues until the MSE of the gbest is lower than the desired threshold or a maximum iteration number is reached. When the iteration is finished, the gbest weights are used as the training results.

Although there has been some research on the application of PSO in recurrent neural network training, its performance is relatively weak. Some of the problems include:

1. The calculation of the outputs of a RNN takes more computational resources. For impedance identification of dynamic systems with RNN, a sequence of sampled voltage and current signals is fed to the network to calculate its output. Normally thousands of data points are needed to accurately represent the system. The evaluation of the cost function is thus computationally expensive. This would put a tight limit on the number of particles in a swarm. Normally 20 to 30 particles are used.
2. The feedback loop in the RNN significantly increases the number of weight parameters. For example, a single-input-single-output two-layered feedforward network with 6 hidden neurons would have 18 weight parameters (including those associated with the bias term). A SISO two-layered recurrent network with the same number of hidden neurons would have 54 weight parameters. This means that the swarm needs to search for an optimal solution in a 54-dimension space. Given the limited number of particles in a swarm, it is very difficult for the swarm to find the optimal solution.

C. Combined PSO-BP training algorithm

Even though both BP and PSO algorithms have limitations in training recurrent neural networks, each of them has its own distinctive advantages.

BP searches the space with the guidance of gradient information of the error surface, which makes the search more efficient than many of the evolutionary methods, especially when the search is around the global minimum.

With proper design of the parameters, PSO can traverse large areas in the search space. Its stochastic characteristic enables it to perform global search in the space without getting trapped in local minima.

A training algorithm that combines both PSO and BP is proposed in this paper. The procedure is as follows:

1. A modified version of the PSO algorithm is first used to train the RNN. The goal of the PSO training is to traverse in a large area of the weight parameter space search for not only the solution with minimal MSE, but also the largest gradient. Finding the final optimal solution is not considered in this stage because it would

take too many iterations and the time consumed might be prohibitive.

2. The modified PSO differs from the standard one in the fifth step. Instead of selecting the gbest solely based on particle MSE, the gradient of each particle is also considered. This is done by first determining the *M* particles with the smallest MSE in the swarm, then picking the one with the largest gradient to compare with gbest. For a 30-particle swarm, *M* is chosen to be 5 in this study.
3. The PSO training terminates when gbest is not updated for some consecutive iterations, or when a maximum iteration number is reached.
4. The training result of the PSO algorithm is then used to initialize a RNN to be trained with the BP algorithm.

V. SIMULATION RESULTS

To demonstrate the effectiveness of the proposed training method, a three-phase inverter system with R-L load is used. As shown in Fig. 4, the inverter is connected to a dc power supply. It converts the dc power into three phase ac power, which is then sent to the R-L load. A PWM controller, which is not shown in the diagram, is used to generate the switching signals for the six IGBT devices. The switching frequency of the PWM signals is 20 kHz, and the modulation index is set to 0.9.

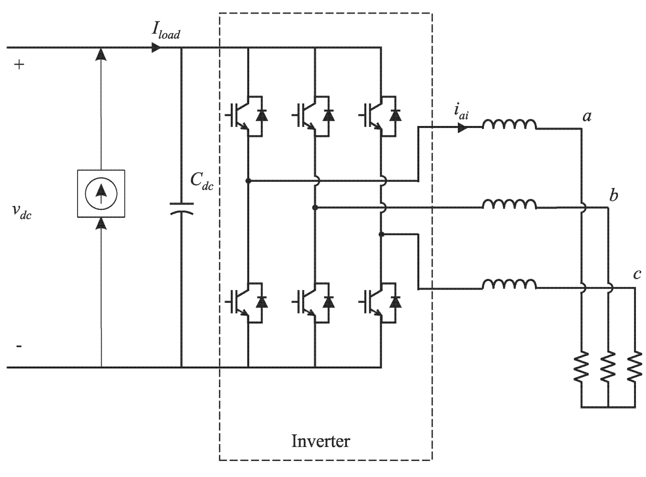

Fig. 4. Diagram of the example inverter system.

For the purpose of system impedance identification, random dc current signals are injected at the input interface of the inverter (the dc link capacitor is included in the inverter model), and the dc voltage and current are recorded for a duration of 0.2 seconds. The sampling frequency is set to be 4 kHz. The signals are normalized to be used in RNN training, as shown in Fig. 5.

The Elman RNN has 6 hidden neurons, and the voltage signal is used as the input, while the current is used as the target output.

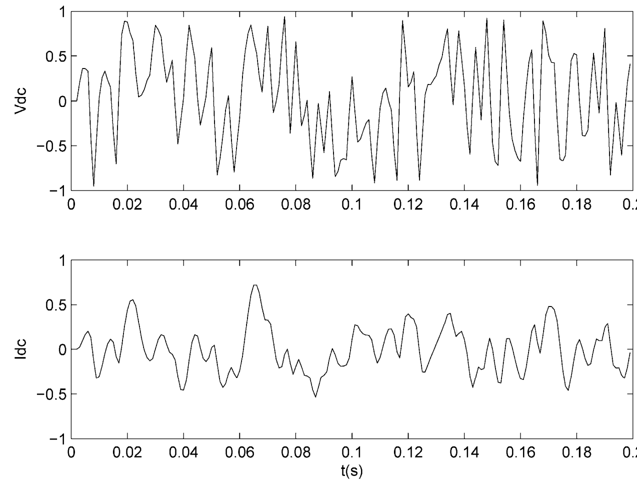

Fig. 5. Training signals used for RNN.

The performance of BP training algorithm is demonstrated by multiple training of the same network with different random initial weights. The maximum epoch number is set to be 500. As can be seen in Fig. 6, after a period of rapid decrease, the MSE gradually settles down. At the end of 500 epochs, most of the MSE curves become almost flat. This indicates that most likely the training has reached a minimal point. However, different initial weights have large impacts on the final MSE values, which range from 1.15e-8 to 0.0196. The final MSE value is very distributed with a mean value of 0.0032. It is evident that some of the training results are trapped in local minima. For example, the top curve converges in less than 100 epochs and the gradient is very close zero.

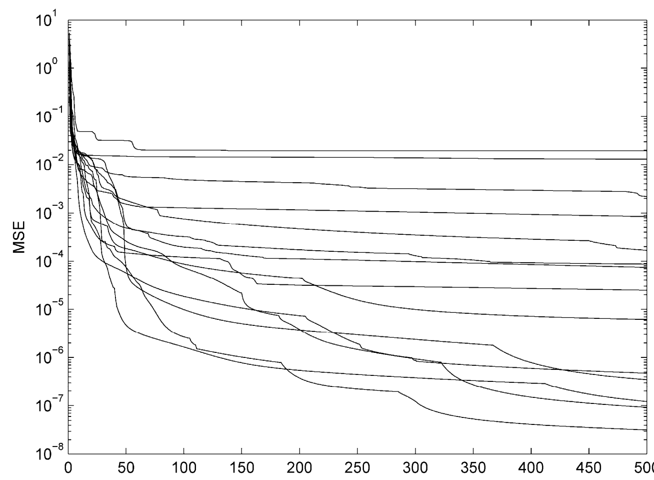

Fig. 6. Training process with BP algorithm.

The performance of standard PSO training is shown in Fig. 7. The PSO parameters are set as follows: w = 0.8, $c_1 = c_2$ = 1.5, N = 30. Due to the high computational efforts involved, the maximum iteration number is set to 100. Comparing the results with those shown in Fig. 6, it can be seen that although the MSE curves are most consistent, the MSE values do not decrease as fast as those in the BP algorithm. This is mainly because the small number of particles in the swarm that limits its search ability. It should be noted that the MSE values are still decreasing at the end of the training, even though at a lower speed. The final MSE values range from 0.025 to 0.054, with a mean value of 0.040.

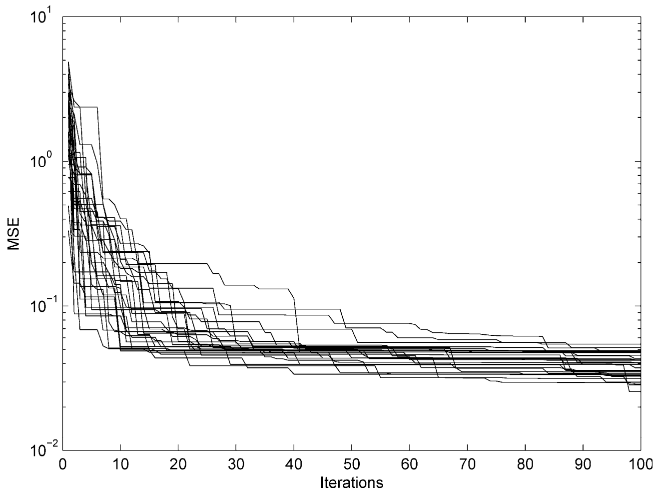

Fig. 7. Training process with PSO algorithm.

Next the combine PSO-BP method is used to train the network. Same PSO parameters are used for the PSO stage, except that the maximum number of iterations is set to be 50. The resulted weights are then used as the initial values for the BP training stage. Fig. 8 shows the MSE curves during the BP training. Because the pre-training in the PSO stage, generally better results can be achieved in the BP stage. Although the modified PSO training cannot guarantee a global minimum can be found in the BP stage, it helps to avoid many local minima by directing the swarm towards areas with not only smaller MSE, but also larger gradient. The final MSE values range from 4.71e-9 to 0.0018, with a mean value of 7.94e-4. It can be seen that both the minimal MSE and mean MSE values are less that those in the training with BP only.

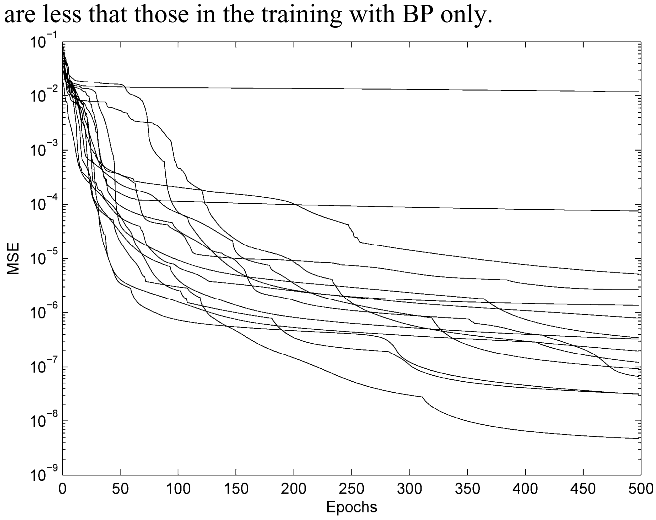

Fig. 8. Training process with combined PSO-BP algorithm.

The trained recurrent neural network is then used as a dynamic model to extract the impedance of the inverter system. The identification procedure involves feeding the neural network with ideal sinusoidal signals and calculating its output. Fourier transform is then used to process the signals and determine the magnitude and phase angle of the system impedance at the specific frequency. Fig. 9 shows the actual and identified impedances of the system in the frequency range of 10 Hz to 1000 Hz. It can be seen that very good agreement is achieved.

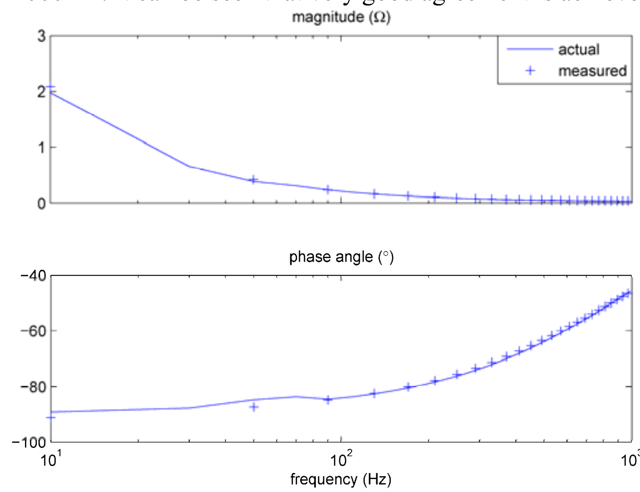

Fig. 9. Actual and identified impedance of the example system.

VI. CONCLUSIONS

In this paper a combined PSO-BP algorithm is used to train a recurrent neural network for the identification of impedances of power electronic systems. The PSO algorithm uses both the MSE and gradient information to guide the movement of the particles. After the neural network is trained with PSO for a number of iterations, BP is used to continue the training on the global best particle found in the PSO training. Simulation results demonstrated that the proposed training algorithm can statistically help avoid the training process being trapped in local minima, without being computationally demanding. Future work includes the determination of optimal division between PSO and BP to achieve a training process with minimal computational resources.

VII. REFERENCES

[1] Barbounis, T.G., Theocharis, J.B., et al., “Long-term wind speed and power forecasting using local recurrent neural network models”, *IEEE Transactions on Energy Conversion*, volume 21, issue 1, pages 273-284, March 2006

[2] Chun-Jung Chen and Tien-Chi Chen, “Design of a Power System Stabilizer Using a new Recurrent Neural Network”, *Innovative Computing, Information and Control*, *ICICIC '06.*, volume 1, pages 39-43 Aug. 2006

[3] Goedtel, A., daSilva, I.N., and Amaral Serni, P.J., “Recurrent Neural Network for Induction Motor Speed Estimation in Industry Applications”, *IEEE MELECON 2006.*, pages 1134-1137, August 2006

[4] Palangpour, P., Venayagamoorthy, G.K. and Duffy, K., “Recurrent Neural Network Based Predictions of Elephant Migration in a South

African Game Reserve", *International Joint Conference on Neural Networks*, pages 4084-4088, July 2006

[5] Familiant, Y.L.; Corzine, K.A.; Huang, J., et. al., "AC Impedance Measurement Techniques", *IEEE International Conference on Electric Machines and Drives*, Pages: 1850 – 1857, May 2005

[6] M. Belkhayat and M.L. Williams, "Impedance Extraction Techniques for Dc and Ac Systems", *Proceedings of the Naval Symposium on Electric Machines*, Philadelphia PA, December 2000.

[7] Janson, D.J. and Frenzel, J.F., "Training product unit neural networks with genetic algorithms", *IEEE Intelligent Systems and Their Applications*, volume 8, issue 5, pages 26-33, Oct. 1993

[8] Alpaydin, G., Dundar, G. and Balkir, S., "Evolution-based design of neural fuzzy networks using self-adapting genetic parameters", *IEEE Transactions on Fuzzy Systems*, volume 10, issue 2, pages 211-221, April 2002

[9] Mendes, R., Cortez, P., et al., "Particle swarms for feedforward neural network training", *Proceedings of the 2002 International Joint Conference on Neural Networks*, volume 2, pages 1895-1899, May 2002

[10] Salerno, J., "Using the particle swarm optimization technique to train a recurrent neural model", *Ninth IEEE International Conference on Tools with Artificial Intelligence*, pages 45-49, Nov. 1997

[11] Gudise, V.G. and Venayagamoorthy, G.K, "Comparison of particle swarm optimization and backpropagation as training algorithms for neural networks", *IEEE Swarm Intelligence Symposium*, pages 110-117, April 2003

[12] Emadi, A., Khaligh, A., Rivetta, C.H., et. al., "Constant power loads and negative impedance instability in automotive systems: definition, modeling, stability, and control of power electronic converters and motor drives", *IEEE Transactions on Vehicular Technology*, Volume 55, Issue 4, Pages:1112 – 1125, July 2006

[13] Middlebrook, R. D., "Input filter considerations in design and application of switching regulators", *IEEE Proceedings of IASAM*, October 1976

[14] Wildrick, C.M., Lee, F.C., Cho, B.H., et. al., "A method of defining the load impedance specification for a stable distributed power system", *IEEE Transactions on Power Electronics*, Volume 10, Issue 3, Pages: 280 - 285, May 1995

[15] Sudhoff, S.D., Glover, S.F., Lamm, P.T., et. al., "Admittance space stability analysis of power electronic systems", *IEEE Transactions on Aerospace and Electronic Systems*, Volume 36, Issue 3, Part 1, Pages:965 - 973, July 2000

[16] Yam, J.Y.F. and Chow, T.W.S., "Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients", *IEEE Transactions on Neural Networks*, volume 12, issue 2, pages 430-434, March 2001

[17] T. Masters, *Practical Neural Network Recipes in C++*. Boston, MA, Academic, 1993.