퍼지 및 지능시스템학회 논문지 2001, Vol. 11, No. 3, pp. 215-222

유전알고리즘을 이용한 신경망 구조 및 파라미터 최적화

Neural Network Structure and Parameter Optimization via Genetic Algorithms

한 승 수
Seung-Soo Han

명지대학교 전기정보제어공학부

요 약

신경망은 선형 시스템 뿐 만 아니라 비선형 시스템에 있어서도 탁월한 모델링 및 예측 성능을 갖고 있다. 하지만 좋은 성능을 갖는 신경망을 구현하기 위해서는 최적화 해야할 파라미터들이 있다. 은닉층의 뉴런의 수, 학습율, 모멘텀, 학습오차 등이 그것인데 이러한 파라미터들은 경험에 의해서, 또는 문헌들에서 제시하는 값들을 선택하여 사용하는 것이 일반적인 경향이다. 하지만 신경망의 전체적인 성능은 이러한 파라미터들의 값에 의해서 결정되기 때문에 이 값들의 선택은 보다 체계적인 방법을 사용하여 구하여야 한다. 본 논문은 유전 알고리즘을 이용하여 이러한 신경망 파라미터들의 최적 값을 찾는데 목적이 있다. 유전 알고리즘을 이용하여 찾은 파라미터들을 사용하여 학습된 신경망의 학습오차와 예측오차들을 심플렉스 알고리즘을 이용하여 찾은 파라미터들을 사용하여 학습된 신경망의 오차들과 비교하여 본 결과 유전 알고리즘을 이용하여 찾은 파라미터들을 이용했을 때의 신경망의 성능이 더욱 우수함을 알 수 있다.

Abstract

Neural network based models of semiconductor manufacturing processes have been shown to offer advantages in both accuracy and generalization over traditional methods. However, model development is often complicated by the fact that back-propagation neural networks contain several adjustable parameters whose optimal values unknown during training. These include learning rate, momentum, training tolerance, and the number of hidden layer neurons. This paper presents an investigation of the use of genetic algorithms (GAs) to determine the optimal neural network parameters for the modeling of plasma-enhanced chemical vapor deposition (PECVD) of silicon dioxide films. To find an optimal parameter set for the neural network PECVD models, a performance index was defined and used in the GA objective function. This index was designed to account for network prediction error as well as training error, with a higher emphasis on reducing prediction error. The results of the genetic search were compared with the results of a similar search using the simplex algorithm.

Key Words : Genetic Algorithm, Neural Network Modeling, Parameter Optimization

1. Introduction

Accurate and efficient modeling of semiconductor fabrication process is necessary for a variety of integrated circuit manufacturing applications, including process control, diagnosis, and yield enhancement. Recently, neural networks have been successfully applied to the modeling of semiconductor fabrication processes such as plasma-enhanced chemical vapor deposition and reactive ion etching [1-3]. Neural networks provide distinctive features that are of potentially revolutionary importance for this application domain. These features include: learning and adaptation, a powerful nonlinear modeling capability, a fine-grained massive parallelism well suited for hardware implementation, and robustness to noise. However, since neural network learning algorithms use only a limited number of examples for a given problem, it is not necessarily guaranteed that the trained network gives correct answers for unknown examples. This presents challenges in developing robust neural process models which are able to generalize beyond the range of their training data.

Multi-layered feed-forward neural networks consist of an input layer, an output layer, and potentially several hidden layers. However, it has been proven theoretically that a three-layer neural network with a sufficient number of hidden units can approximate any continuous function with an arbitrarily small error [4]. The number

접수일자 : 2000년 11월 24일
완료일자 : 2001년 4월 23일
감사의 글 : 본 연구는 명지대학교 신임교수 연구정착금 지원사업에 의해 일부 지원 받았습니다.

of hidden neurons must be large enough to form a decision region of sufficient complexity for a given problem, but it is not clear how many hidden units is optimal for maximum generalization. Aside from the network architecture, there are also learning parameters in back-propagation neural network training whose optimal values are critical to the success of the learning process but unknown prior to training. These include such variables as the learning rate, momentum, and training tolerance. In order to increase the benefits of the neural network based process modeling strategy, a systematic means of selecting an optimal network structure and set of learning parameters is an essential requirement.

Several previous efforts at obtaining the optimal neural network structure and parameters have been described in the literature. Fogel derived a modified version of Akaike's information criterion called the final information statistic (FIS) to select the optimal neural network structure [5]. Wada and Kawato proposed a statistically based information criterion which can be used to find optimal number of hidden units to maximize the generalization capability of 3-layer networks [6]. However, each of the aforementioned efforts at neural network optimization have focused on improving network performance in binary classification and pattern recognition tasks.

Kim and May investigated the effects of structural and learning parameters on the performance of a neural network designed to perform function approximation [7]. By using a statistically designed experiment, they varied hidden layers, number of hidden neurons per layer, learning rate, momentum, initial weight range, and training tolerance. Optimal values for these variables were determined using the Nelder-Mead simplex search algorithm [8]. However, the simplex method is fundamentally dependent on its initial search point. With an improper starting point, overall performance degrades and this algorithms is likely to be trapped in local optima.

In the 1970s, Holland introduced so-called "genetic" algorithms (GAs) as an alternative search and optimization procedure to traditional calculus based "hill-climbing" methods [9]. GAs refer to a family of computational models inspired by evolution. Theoretical analyses suggest that they can quickly locate high performance regions in extremely large and complex search space and possess some natural insensitivity to noise. These attributes also make GAs a very attractive technique for determining optimal neural network structure and learning parameters. Harp and Samad developed a package called NeuroGENESYS to optimize neural network structure for small scale pattern recognition problems using the genetic approach, but only compared their results to control studies using random search [10]. Dodd also used genetic techniques to optimize neural networks [11].

In this paper, we apply genetic algorithms to search for the optimal neural network structure and learning parameters in modeling the semiconductor fabrication process of plasma enhanced chemical vapor deposition (PECVD). The goal is to design an optimal neural network for a specific semiconductor manufacturing problem: modeling the PECVD of silicon dioxide films in as a function of gas flow rates, temperature, pressure and RF power. The responses modeled include film permittivity, refractive index, residual stress, uniformity, and impurity concentration. To obtain the necessary training data for developing the optimal neural process model, a fractional factorial experiment has been performed to simultaneously investigate the effect of the number of hidden layer neurons, training tolerance, learning rate and momentum [2]. The objective of the experiment is to derive an optimal set of parameters for a given set of performance metrics.

The network responses which are optimized are learning capability and predictive capability, which are quantified by a performance index which accounts for the mean-squared error of the model. Optimal parameter sets have been determined which minimize learning and prediction error have been determined using genetic search, and this technique is compared with the simplex method. The genetic algorithm optimization procedure significantly outperformed the simplex search, yielding approximately 10% improvement in network training error and 65% improvement in prediction error.

2. Neural Process Modeling and Experimental Design

2.1 PECVD SiO2 Film Characterization

The silicon dioxide films were deposited in a Plasma-Therm 700 series batch reactor using nitrous oxide, 2% silane in nitrogen, and nitrogen as feed gases. The deposition conditions, shown in Table 1, were varied in a central composite circumscribed design [13]. The central composite design employed consisted of a 25-1 fractional factorial augmented by ten axial points and three center points.

Table 1. Deposition Parameters

Parameter	Range
Substrate Temperature	200 - 400 ℃
Pressure	0.25 - 1.8 torr
RF Power	20 - 150 watt
2% SiH4 in N2 Flow	200 - 400 sccm
N2O Range	400 - 900 sccm

Approximately five microns of SiO2 were deposited on 4" diameter (100) oriented silicon wafers. In addition to the twenty-nine experiments which provided data to train the neural process models, eight other experimental runs were performed to test the performance of the models.

After deposition, a Metricon 2010 prism coupler was used to determine the thickness and index of refraction of the films on the wafer. These were measured at five points around the wafer to examine film uniformity. A Flexus F2320 was used to measure the change in radius of curvature of the bare silicon substrate due to the stress induced by the grown silicon dioxide film. A Perkin-Elmer 1600 FTIR was used to obtain the infrared spectra, which were used to measure the impurity (H2O and SiOH) content of the films. Parallel-plate capacitors were fabricated to evaluate film permittivity, and a Keithley 590 CV analyzer and HP 4275 LCR meter were used to measure the film capacitance.

2.2 Neural Network Structure and Learning Parameters

Neural networks possess the capability of learning arbitrary nonlinear mappings between noisy sets of input and output patterns. Neural network learning is designed to determine an appropriate set of connection strengths which allow many simple parallel processing units to achieve a desired state of activation that mimics a given set of sampled patterns. These rudimentary processors (called "neurons") are interconnected in such a way that knowledge is stored in the weight of the connections between them. The activation level of a neuron is determined by a nonlinear "activation function." This activation function endows the network with the ability to generalize with an added degree of freedom not available in statistical regression techniques [1].

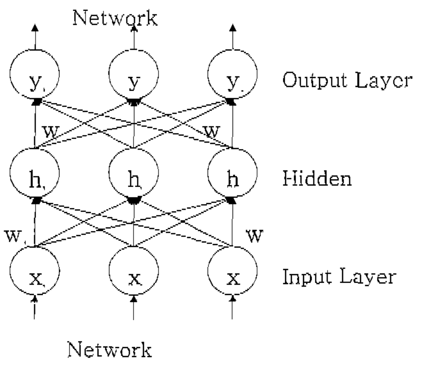

Figure 1. Feed-Forward Neural Network

Neural networks used for semiconductor process modeling are typically trained via the error back-propagation (BP) algorithm. Figure 1 depicts the general structure of a feed-forward, multi-layered neural network. Individual neurons in the network receive, process, and transmit critical information regarding the relationships between input and output pairs. The input layer of neurons corresponds to the five adjustable input parameters which are varied in the PECVD experiment. The output layer corresponds to the deposition variables to be modeled. The network also incorporates one "hidden" layer of neurons which do not interact with the external world, but assist in performing classification and feature extraction on information provided by the input and output layers. Conceptually, the hidden layer neurons can be viewed as representing fundamental, yet not directly controllable plasma properties such as electron temperature or reactive species concentration.

In BP training, model performance is influenced by both the number of hidden layers and the number of neurons in each layer. It has been shown that a BP network containing a single hidden layer can encode any arbitrarily complex input-output relationship [4]. In the process of finding the optimal network structure, the number of layers is therefore fixed at three, and only the number of neurons in the hidden layer is varied. Hidden neurons provide an estimate of the number of "conflicts" contained in the input/output mapping (where a conflict refers to mappings which require incompatible weight solutions) [13]. A large number of hidden neurons is required to model complex relationships, but too many can result in an over-trained network and can render it incapable of generalizing input/output relationships that differ from the training samples [5].

The back-propagation procedure uses a gradient descent technique, which systematically changes the network weights by an amount proportional to the partial derivative of the accumulated error function, E, with respect to the given weight. In other words, the change in weight, Δw_{ijk}, is given by:

$$\Delta w_{ijk} = -\eta \frac{\partial E}{\partial w_{ijk}} \tag{1}$$

where i denotes a node in layer k, and j a node in the preceding layer (k-1), and wijk the weight between these two nodes. The key to the back-propagation procedure is that the above partial derivative can be computed using the chain rule from the derivative of the system error with respect to actual output of the network, the derivative of the output of each neuron with respect to its input, and finally the derivative of the neural inputs with respect to the weights.

The constant η is called the learning rate, where $0 < \eta < 1$. The learning rate determines the speed of convergence by regulating the step size. However, the network may venture too far from the actual minimum value of the error surface or oscillations in the error function may occur during training if h gets too large [14]. On the other hand, smaller rates can ensure the stability of the network by diminishing the gradient of noise in the weights, but result in longer training time.

One improvement to the standard weight modification technique that has been suggested is to expand Equation

(1) by adding a momentum term:

$$\Delta w_{ijk}(n+1) = -\eta\frac{\partial E}{\partial w_{ijk}} + \alpha\Delta w_{ijk}(n) \tag{2}$$

where α is usually taken to be $0 \leq \alpha < 1$, and n is the number of times a pattern has been presented to the network. The effects of the momentum term are to magnify the learning rate for flat regions of weight space where the gradients are more or less constant, and to prevent oscillations. This additional term, computed by adding a fraction of the previous weight change, tends to keep the weight changes going in the same direction. Both the rate and the existence of network convergence depend on the proper selection of the learning rate and of the momentum factor [7].

Aside from learning rate and momentum, another critical learning parameter is network training tolerance. Training tolerance determines the overall quality of the network modeling capability by specifying the accuracy of the neural outputs. A smaller training tolerance usually increases learning accuracy, but can result in less generalization capability as well as longer training time. Conversely, a larger tolerance enhances convergence speed at the expense of accuracy in learning

3. Optimization of Network Parameters

3.1 Network Optimization Using the Simplex Method

For the PECVD process models, an effort was undertaken to optimize each of the aforementioned BP learning parameters, as well as the network structure [2]. This was done to maximize the accuracy of each model by minimizing its training and prediction error. In all, four parameters are considered: number of hidden neurons, learning rate, momentum, and training tolerance. Initially, the neural PECVD models were obtained using a default network structure and set of learning parameters. These "rough" models were then refined by varying the values of the four critical network parameters according to a 24 factorial design with three center point replications. The experimental ranges of each parameter and their default values are summarized in Table 2.

Table 2. Ranges of Neural Network Parameters

Parameters	Range	Default Value
No. of Hidden Neurons	3 - 9	6
Learning Rate	0.05 - 0.5	0.275
Momentum	0.35 - 0.95	0.65
Training Tolerance	0.01 - 0.13	0.07

The results of the fractional factorial experiment were analyzed using the commercial statistical software package, RS/Discover [15]. RS/Discover used the data from the fractional factorial experiment to construct a regression model which predicted the value of a performance index for the neural network models as a function of the four network parameters listed in Table 2. The performance index (PI) is given by:

$$PI = K_1\sigma_t^2 + K_2\sigma_p^2 \tag{3}$$

where σ_t is the RMS network training error and σ_t is the RMS network prediction error. The constants K1 and K2 are weights representing the relative importance of each performance measure. Since the prediction error is typically the more important quality characteristic, the values chosen for these constants were K1 = 1, and K2 = 10.

Since the RS/Discover package employs the Nelder-Mead simplex algorithm [16] for optimization, this algorithm was used in [2] to minimize the PI. A regular simplex is defined as a set of (n+1) mutually equidistant points in n dimensional space. The main idea of the simplex method is to compare the values of the function to be optimized at the (n+1) vertices of the simplex and move the simplex towards the optimal point iteratively. The original simplex method maintained a regular simplex at each stage. Nelder and Mead proposed several modifications to the method which allow the simplices to become non-regular. The result is a very robust direct search method which is extremely powerful, provided that the number of variables does not exceed five or six. The result of minimizing the network performance index using the simplex method will be compared to optimization using genetic algorithms, which are described in greater detail below.

3.2 Genetic Optimization of Network Parameters

Genetic Algorithms (GAs) are guided stochastic search techniques based on the mechanics of genetics [9]. They use three genetic operations found in natural genetics to guide their trek through the search space: reproduction, crossover, and mutation [17-18]. Using these operations, GAs are able to search through large, irregularly shaped spaces quickly, requiring only objective function value information (detailing the quality of possible solutions) to guide the search. This is a desirable characteristic, considering that the majority of commonly used search techniques require derivative information, continuity of the search space, or complete knowledge of the objective function to guide their search. Furthermore, GAs take a more global view of the search space than many methods currently encountered in engineering optimization.

In computing terms, a genetic algorithm maps a problem on to a set of binary strings, each string representing a potential solution. The GA then manipulates the most promising strings in searching for improved solutions. A GA operates typically through a simple cycle of four stages:

I) Creation of a "population" of strings,
ii) Evaluation of each string,
iii) Selection of "best" strings, and
iv) Genetic manipulation, to create the new population of strings.

In each computational cycle, a new generation of possible solutions for a given problem is produced. At the first stage, an initial population of potential solutions is created as a starting point for the search process. Each element of the population is encoded into a string (the "chromosome"), to be manipulated by the genetic operators. In the next stage, the performance (or "fitness") of each individual of the population is evaluated with respect to the constraints imposed by the problem. Based on each individual string's fitness, a selection mechanism chooses "mates" for the genetic manipulation process. The selection policy is responsible for assuring survival of the most "fit" individuals.

In coding for a genetic search, binary strings are typically used, although alphanumeric strings can be used as well. The method of coding multi-parameter optimization problems used in this paper is concatenated, multi--parameter, mapped and fixed-point coding [17]. If x ∈ [0, 2l] is the parameter of interest (where l is the length of the string), the decoded unsigned integer x can be mapped linearly from [0, 2l] to a specified interval [Umin, Umax]. To construct a multi-parameter coding, as many single parameters as required can simply be concatenated. Each coding may have its own sub-length (i.e.- its own Umin and Umax). Figure 2 shows an example of a 2-parameter coding with four bits in each parameter. The ranges of the first and second parameter are 2-5 and 0-15, respectively.

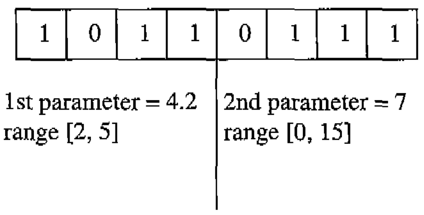

Figure 2. Multi-Parameter Coding

GA consists of three operations, namely reproduction, crossover and mutation. Reproduction is the process by which strings with high fitness values (i.e. - good solutions to the optimization problem under consideration) receive appropriately large numbers of copies in the new population. The reproduction method selected for neural network optimization is elitist roulette wheel selection [18]. In this method, those strings with large fitness values Fi are assigned a proportionately higher probability of survival into the next generation. This probability distribution is determined according to:

$$P_{select\ i} = \frac{F_i}{\sum F} \tag{4}$$

Thus, an individual string whose fitness is n times better than another will produce n times the number of offspring in the subsequent generation. Once the strings have reproduced, they are stored in a "mating pool" awaiting the actions of the crossover and mutation operators.

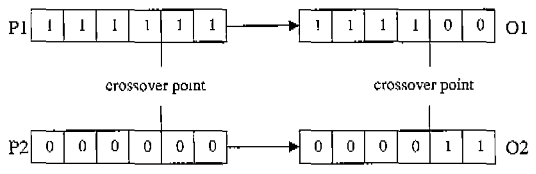

Figure 3. The Crossover Operation

The crossover operator takes two chromosomes and interchanges part of their genetic information to produce two new chromosomes (see Figure 3). After the crossover point has been randomly chosen, portions of the parent strings (P1 and P2) are swapped to produce the new offspring (O1 and O2) based upon a specified crossover probability. Mutation is motivated by the possibility that the initially defined population might not contain all of the information necessary to solve the problem. This operation is implemented by randomly changing a fixed number of bits every generation based upon a specified mutation probability (see Figure 4). Typical values for the probabilities of crossover and bit mutation range from 0.6 to 0.95 and 0.001 to 0.01, respectively. In this paper, the probabilities of crossover and mutation were set to 0.6 and 0.01, respectively.

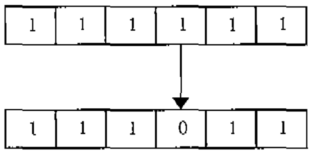

Figure 4. The Mutation Operation

Our overall neural network optimization scheme is shown in Figure 5. Genetic algorithms generate possible candidates for optimal neural parameters using an initial population of 50 potential solutions as a starting point for the search process. Each element of the population is encoded into a 10 bit string (the "chromosome"), to be manipulated by the genetic operators. Since there are four parameters to be optimized (number of hidden layer neurons, momentum, learning rate, and training tolerance), the concatenated total chromosome length is a 40 bit string.

The performance (or "fitness") of each individual of the population is evaluated with respect to the constraints imposed by the problem based on the fitness function. To search for parameter values which minimized both network training error and prediction error, the performance index (PI) in Equation (3) was once again implemented. For genetic optimization of the neural network models, the desired output is reflected by the following fitness function (F):

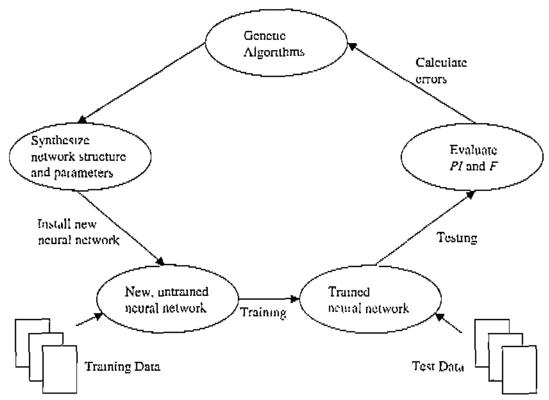

Figure 5. Block Diagram of Genetic Optimization

$$F = \frac{1}{1+PI} \tag{5}$$

Maximization of F continues until a final solution is selected after 100 generations. If the optimal solution is not found after 100 generations, the solution with the best fitness value is selected.

4. Results and Discussion

4.1 Network Optimization for Individual PECVD Responses

Individual response neural network models were trained to predict PECVD silicon dioxide permittivity, refractive index, residual stress, and non-uniformity, and impurity (H2O and SiOH) concentration. The result of genetic optimization of these neural process models is shown in Table 3. Analogous results for network optimization using the simplex method are given in Table 4.

Table 3. Network Parameters Optimized by Genetic Algorithms

PECVD Response	Hidden Neurons	Momen-tum	Learning Rate	Training Tolerance
Permittivity	7	0.41	0.19	0.01
Ref. Index	7	0.40	0.37	0.08
Stress	7	0.39	0.07	0.06
Non-Uniformity	4	0.43	0.06	0.11
Impurity Concentration	4	0.37	0.08	0.07

Table 4. Network Parameters Optimized by Simplex Search

PECVD Response	Hidden Neurons	Momen-tum	Learning Rate	Training Tolerance
Permittivity	9	0.40	0.50	0.01
Ref. Index	9	0.40	0.50	0.01
Stress	6	0.40	0.05	0.13
Non-Uniformity	6	0.40	0.50	0.13
Impurity Concentration	6	0.40	0.05	0.07

Examination of Tables 3 and 4 shows that the most significant differences between the two optimization algorithms occur in the number of hidden neurons and learning rates predicted to be optimal. Genetic search generally leads to fewer hidden neurons and smaller learning rates. Momentum and training tolerance values are comparable for both methods.

Tables 5 and 6 compare the normalized training error (σ_t) and prediction error (σ_p) for the two search methods, respectively. (In each table, the "% improvement" column refers to the improvement obtained by using genetic search). Although in two cases involving training error minimization the simplex method proved superior, the genetically optimized networks exhibited vastly improved performance in nearly every category for prediction error minimization. The overall average improvement observed in using genetic optimization was 1.6% for network training error and 60.4% for prediction error. These observations indicate

Table 5. Training Error Comparison of Simplex and GAs Network Optimization

PECVD Response	Simplex	GAs	%Improvement
Permittivity	0.0578	0.0110	80.94
Ref. Index	0.0232	0.0822	-71.76
Stress	0.0500	0.0571	-12.47
Non-Uniformity	0.1146	0.1099	4.09
Impurity Concentration	0.0951	0.0841	7.35

Table 6. Prediction Error Comparison of Simplex and GAs Network Optimization

PECVD Response	Simplex	GAs	%Improvement
Permittivity	0.1788	0.0363	79.68
Ref. Index	0.2158	0.0591	72.61
Stress	0.9659	0.4815	50.16
Non-Uniformity	0.1361	0.0246	81.92
Impurity Concentration	0.0964	0.0795	17.54

that the search spaces provided by the neural PECVD model parameters are generally multimodal, rather than unimodal. The simplex method tends to become trapped in local optima in multimodal search spaces.

4.2 Network Optimization for Multiple a PECVD Response Model

The parameter sets called for in Tables 3 and 4 are useful to obtain optimal performance for a single PECVD response, but can provide suboptimal results for the remaining responses. For example, Table 3 indicates that seven hidden neurons are optimal for permittivity, refractive index, and stress, but only four hidden neurons are necessary for the non-uniformity and impurity concentration models. Obviously, it is more desirable to optimize network parameters for all responses simultaneously. Therefore, a multiple output neural process model (which includes permittivity, stress, non-uniformity, H2O and SiOH) was trained with that objective in mind.

Table 7 shows the optimized network parameters for the multiple response PECVD model. Here, the optimal parameter sets derived by genetic and simplex search differ only slightly. However, they differ noticeably from the parameter sets optimized for individual PECVD responses. This is especially true for the number of hidden neurons and the learning rates.

The seemingly slight differences in optimal network parameters can nevertheless lead to significant differences in network performance. This is indicated in Table 8, which shows the training and prediction errors for the neural network models trained using the parameter sets in Table 7. Clearly, the genetic search yields superior results in both training and prediction. If the improvements in performance for the multiple PECVD response model is factored in, GAs provide an average benefit of 10.0% in network training accuracy and 65.6% in prediction accuracy.

Table 7. Optimal Network Parameters for Multiple Response PECVD Models

Network Parameters	Simplex	GAs
Hidden Neuron	5	5
Momentum	0.35	0.37
Learning Rate	0.05	0.08
Training Tolerance	0.13	0.06

Table 8. Error Comparison for Optimal Multiple Response PECVD Models

Error	Simplex	GAs	%Improvement
σt	0.2891	0.1391	51.88
σp	1.2979	0.1104	91.50

5. Conclusion

Although back-propagation neural networks have been successfully applied to the modeling of semiconductor fabrication processes, their performance is largely dependent on network structure and learning parameters. In this paper, genetic algorithm based optimization of neural network structure and learning parameters is proposed. Genetic search is less likely to be trapped in local extrema than other optimization techniques. This genetic optimization approach was shown to lead to improvements over the simplex algorithm in selecting optimal neural network structure and learning parameters. Overall, the genetic search outperformed simplex method by approximately 10% in reducing training error and by about 66% in reducing prediction error. The genetic based approach is therefore shown to be a robust and powerful search method in selection of optimal neural network parameters sets.

References

[1] C. Himmel, and G. May, "Advantages of Plasma Etch Modeling Using Neural Networks Over Statistical Techniques," *IEEE Trans. Semi. Manufac.*, vol. 6, pp. 103-111, May, 1993.

[2] S. Han, M. Ceiler, S. Bidstrup, P. Kohl, and G. May, "Modeling the Properties of PECVD Silicon Dioxide Films Using Optimized Back-Propagation Neural Networks," *IEEE Trans. Comp. Pack. Manufac. Tech. Part A*, vol. 17, no. 2, pp. 174-182, June, 1994.

[3] F. Nadi, A. M. Agogino, and D. A. Hodges, "Use of Influence Diagrams and Neural Networks in Modeling Semiconductor Manufacturing Processes," *IEEE Trans. Semi. Manufac.*, vol. 4, no. 1, pp. 52-58, Feb. 1991.

[4] B. Irie and S Miyake, "Capabilities of Three-layered Perceptrons," *IEEE International Conference on Neural Networks*, pp. 641-648, 1988.

[5] D. Fogel, "An Information Criterion for Optimal Neural Network Selection," *IEEE Trans. Neural Networks*, vol. 2, no. 5, pp. 490 497, Sept. 1991.

[6] Y. Wada and M. Kawato, "Estimation of Generalization Capability by Combination of New Information Criterion and Cross Validation," *Proc. Int. Joint Conf. Neural Networks*, vol. 2, 1991, pp. 1-6.

[7] B. Kim and G. May, "An Optimal Neural Network Process Model for Plasma Etching," *IEEE Trans. Semi. Manufac.*. vol. 7, pp. 12-21, Feb., 1994.

[8] J. C. Nash, Compact Numerical Methods for Computers, New York: Wiley, 1979.

[9] J. H. Holland, Adaptation in Natural and Artificial Systems, Ann Arbor, MI, University of Michigan Press, 1975.

[10] S. A. Harp and T. Samad, "Optimizing Neural Networks with Genetic Algorithms," *Proc. American Power Conference*, vol. 2, 1992, pp. 1138-1143.
[11] N. Dodd, "Optimisation of Network Structure using Genetic Techniques," *Proc. Internat'l Joint Conf. Neural Networks*, vol. 3, 1990, pp. 965-970.
[12] G. Box, W. Hunter, and J. Hunter, Statistics for Experimenters, New York: Wiley, 1978.
[13] M. Gutierrez, J. Wang, and R. Grondin, "Estimating Hidden Units for Two-Layer Perceptrons," *Proc. IEEE Internat'l Conf. on Artifical Neural Networks*, 1989, pp. 120-124.
[14] J. Cater, "Successfully Using Peak Learning Rates of 10 (and greater) in Back-Propagation Networks with the Heuristic Learning Algorithm," *Proc. IEEE Internat'l Conf. on Artificial Neural Networks*, vol. 2, 1987, pp. 645-651.
[15] RS/Discover User's Guide, BBN Software Products, 1992.
[16] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, vol. 7, pp. 308-313, 1965.
[17] D. E. Goldberg, Genetic Algorithms in Search, Optimization & Machine Learning, Addison Wesley, 1989.
[18] J. F. Frenzel, "Genetic Algorithms," *IEEE Potentials*, pp. 21-24, Oct. 1993.

저 자 소 개

한승수 (Seung-Soo Han)
1986년 : 연세대학교 전기공학과 졸업.
1988년 : 연세대학교 전기공학과 석사
1986년 : 조지아공대 박사

관심분야 : 신경회로망, 유전알고리즘, DNA Computing, 정보보호

Phone : (031) 330-6345
Fax : (031) 321-0271
E-mail : shan@mju.ac.kr