A Fuzzy Neural Network for Intelligent Data Processing

Wei Xie[1], Feng Chu[2], Lipo Wang[2] , Eng Thiam Lim[1]
[1]Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613
[2]Nanyang Technological University, Block S1, Nanyang Avenue, Singapore 639798
Email: wxie@i2r.a-star.edu.sg

ABSTRACT

In this paper, we describe an incrementally generated fuzzy neural network (FNN) for intelligent data processing. This FNN combines the features of initial fuzzy model self-generation, fast input selection, partition validation, parameter optimization and rule-base simplification. A small FNN is created from scratch - there is no need to specify the initial network architecture, initial membership functions, or initial weights. Fuzzy IF-THEN rules are constantly combined and pruned to minimize the size of the network while maintaining accuracy; irrelevant inputs are detected and deleted, and membership functions and network weights are trained with a gradient descent algorithm, i.e., error backpropagation. Experimental studies on synthesized data sets demonstrate that the proposed Fuzzy Neural Network is able to achieve accuracy comparable to or higher than both a feedforward crisp neural network, i.e., NeuroRule, and a decision tree, i.e., C4.5, with more compact rule bases for most of the data sets used in our experiments. The FNN has achieved outstanding results for cancer classification based on microarray data. The excellent classification result for Small Round Blue Cell Tumors (SRBCTs) data set is shown. Compared with other published methods, we have used a much fewer number of genes for perfect classification, which will help researchers directly focus their attention on some specific genes and may lead to discovery of deep reasons of the development of cancers and discovery of drugs.

KEY WORDS: Fuzzy neural network, data mining, data classification.

1. INTRODUCTION

Intelligent data analysis is an interdisciplinary study concerned with the effective analysis of data. The pervasive use of sensor and information technology has resulted in the generation of vast amounts of digital data at a rapidly increasing rate. Converting raw data into useful information for human decision makers is one of the driving forces behind research into applications of data mining.

Data mining is one of the most promising fields for application of intelligent information processing technologies. A main aim of data mining is to extract useful patterns and nontrivial relationships from large collections of data records, and provide users with a powerful tool for exploiting a great amount of stored data. Data mining is an immensely heterogeneous research area that embraces techniques and ideas that stem from probability and statistics, neuro-computing, rough sets, fuzzy sets, data visualization, databases, and so forth. In spite of such a profound diversity, the focal point is constant: to reveal patterns that are not only meaningful but also easily comprehensible. The requirement forces us to represent data and use algorithms that are conducted at a certain level of information granularity, rather than being confined exclusively to tedious number crunching. Hence here we concentrate on the technology of fuzzy logic in intelligent data processing because it provides a highly intuitive and appealing presentation to the end user.

In the literature, crisp neural networks often have a fixed architecture, i.e., a pre-determined number of layers with pre-determined numbers of neurons. The weights are usually initialized to small random values. Knowledge-based networks[4,11] use crude domain knowledge to generate the initial network architecture. This helps in reducing the

Data Mining, Intrusion Detection, Information Assurance, and Data Networks
Security 2005, edited by Belur V. Dasarathy, Proceedings of SPIE Vol. 5812
(SPIE, Bellingham, WA, 2005) · 0277-786X/05/$15 · doi: 10.1117/12.603175

search space and time required for the network to find an optimal solution. There have also been mechanisms to generate crisp neural networks from scratch, i.e., initially there are no neurons or weights which are generated and then refined during training. For example, Mezard and Nadal's tiling algorithm[9], Fahlman and Lebiere's cascade-correlation[3], and Giles et al's constructive learning of recurrent networks[5] are very useful.

For FNNs, it is also desirable to shift from the traditional fixed architecture design methodology[13] to self-generating approaches. Higgins and Goodman[6] proposed an algorithm to create an FNN according to input data. New membership functions are added at the point of maximum error on an as-needed basis, which will be adopted in the present paper. Juang and Lin[14] proposed a self-constructing FNN with on-line learning. New membership functions are added based on input-output space partitioning using a self-organizing clustering algorithm. Chao and Chen[2] applied fuzzy similarity measure to eliminate redundant fuzzy logic rules.

Frayman and Wang[15] proposed a FNN using fuzzy similarity measure to remove irrelative inputs and rule applicability coefficient was used for rule-base simplification. We proposed a new FNN, with much simpler and effective input selection and rule-base simplification abilities.

The paper is organized as follows. Section 2 describes the implementation of our FNN. Some experimental results on synthesized data classification using our FNN and comparisons with the pruned feedforward crisp neural network and decision tree approaches are shown in section 3. In Section 4, our FNN is applied to solve cancer classification problem based on gene microarray data. Section 4 concludes the paper.

2. IMPLEMENTATION OF OUR FNN

The structure of our FNN is shown in Fig. 1. The network consists of four layers, i.e., the input layer, the input membership function layer, the rule layer, and the output layer[15].

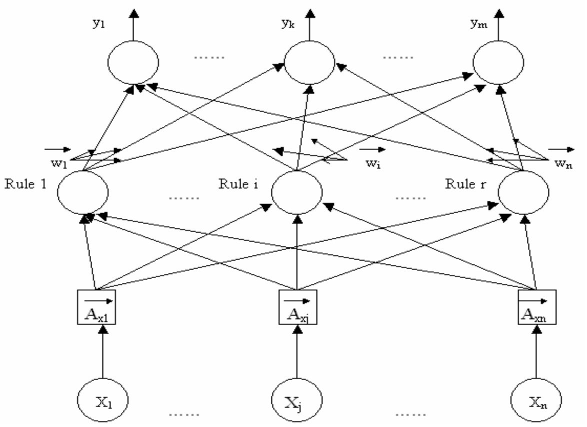

Figure 1: Structure of FNN

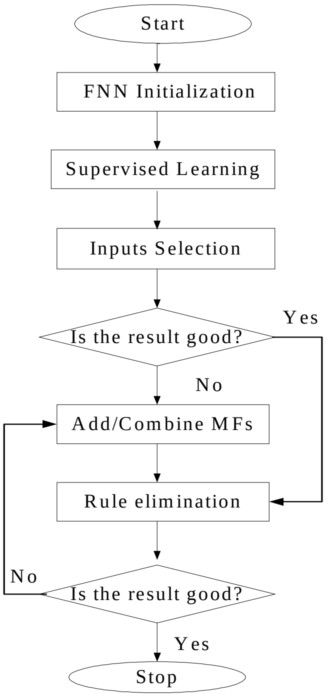

Figure 2: Flow chart of the FNN algorithm

In databases, data fields are either numerical or categorical. The input membership function layer generates input membership functions for numerical inputs, i.e., numerical values are converted to categorical values.

Each rule node is connected to all input membership function nodes and output nodes for this rule. Each rule node performs a product of its inputs. The input membership functions act as fuzzy weights between the input layer and the rule layer. Links between the rule layer, the output layer and the input membership functions are adaptive during learning. In the output layer each node receives inputs from all rule nodes connected to this output node and produces the actual output of the network.
The structure generation and learning algorithm of the FNN are as follows in Fig. 2.

2.1 FNN Initialization

Firstly we create n nodes for the input layer and m nodes for the output layer, where n and m are the number of the input variables (attributes) and the output variables (classes), respectively. The rule layer is empty, i.e., there are initially no rules in the rule base [15].
Two equally spaced triangular membership functions are added along the operating range of each input variable. In such a way these membership functions will satisfy ε-completeness. Piecewise-linear triangular membership function is chosen for computational efficiency [6].

Then we create the initial rule base layer using the following form for rule *i*:

$$\begin{array}{lll} \text{Rule } i: & \text{IF} & x_1 \text{ is } A^i_{x1} \text{ and } \ldots\ x_n \text{ is } A^i_{xn} \\ & \text{Then} & y_1 = \omega_1^i, \ldots, y_m = \omega_m^i, \qquad \text{----} \quad (1) \end{array}$$

where x_j (j=1,2,.., n), and y_l (l=1,2,.., m) are the inputs and the outputs, respectively. ω_l^i is a real number. A^i_q (q=x_1, x_2,… x_n) is the membership function of the antecedent part of rule *i* for node *q* in the input layer[16].

The membership value μ_i of the premise of the *i*th rule is calculated as fuzzy AND, using the product operator

$$\mu_i = A_{x1}{}^i(x_1) \times A_{x2}{}^i(x_2) \times \cdots \times A_{xn}{}^i(x_n) \quad \text{---- (2)}$$

The output y_l of the fuzzy inference is obtained using the weighted average[17].

$$y_l = \frac{\sum_i \mu_i \times \omega_l{}^i}{\sum_i \mu_i} \qquad \text{----(3)}$$

2.2 FNN Training

The network is trained using the following general learning rule[18].

$$y_l{}^i(k+1) = y_l{}^i(k) - \eta \frac{\partial \varepsilon_l}{\partial y_l{}^i} \qquad \text{----(4)}$$

The learning rules for ω_l^i and A^i_j are:

$$\omega_l{}^i(k+1) = \omega_l{}^i(k) - \eta \frac{\partial \varepsilon_l}{\partial \omega_l{}^i} \qquad \text{----(5)}$$

$$A_q{}^i(k+1) = A_q{}^i(k) - \eta \frac{\partial \varepsilon_l}{\partial A_q{}^i} \qquad \text{----(6)}$$

where η is the learning rate. The objective is to minimize an error function

$$\varepsilon_l = \frac{1}{2} \times (y_l - y_{dl})^2 \qquad \text{----(7)}$$

where y_l is the current output, and y_{dl} is the target output.

We let the learning rate η vary to improve the speed of convergence, as well as the learning performance (accuracy). We update η according to the following two heuristic rules:

1) If the error measure undergoes five consecutive reductions, increase η by 5%.
2) If the error measure undergoes three consecutive combinations of one increase and one reduction, decrease η by 5%.

Furthermore, due to this dynamical update strategy, the initial value of η is usually not critical as long as it is not too large.

The learning error ε_l is reduced towards zero or a pre-specified small value $\varepsilon_{def} > 0$ as the iteration number k increases.

2.3 Input Selection

Based on the initial fuzzy model that incorporates all possible input variables, we can evaluate the importance of each input variable. The objective of this work is to reduce the input dimensionality of the model without significant loss in accuracy. Elimination of redundant input features may even improve accuracy.

It is known that the change of system output is contributed to by all input variables. The larger the output change caused by a specified input variable, the more important this input may be. The fuzzy inference system provides an easy mechanism to test the importance of each input variable without having to generate new models[7]. The basic idea is to let the antecedent clauses associated with a particular input variable *i* in the rules be assigned a truth-value of 1 and then compute the fuzzy output, which is due to the absence of input *i*. Then we rank those fuzzy outputs from worst to best, and the worse result indicates the associated input is most important.

Further more, we have to decide how many inputs should be selceted. We start from using the most important input as the only input to the FNN, setting antecedents associated with all other inputs to 1. In the following steps, each time we add one more input according to the ranking order. The subset with best output result is selected as the inputs group; other inputs and associated membership functions are deleted. This approach is proved to be very simple, fast and effective.

2.4 Partition Validation

If the degree of overlapping of membership functions is greater than a threshold, we need to combine them[2]. We use the following *fuzzy similarity measure*[19]

$$\text{E } (A_1, \text{A2}) = \frac{M(A_1 \cap A_2)}{M(A_1 \cup A_2)} \qquad \text{----(8)}$$

where ∩ and ∪ denote the intersection and union of two fuzzy sets A_1 and A_2, respectively. M (•) is the size of a fuzzy set, and $0 \leq$ E (A_1, A2) ≤ 1.

From the formula above, we find that the computation of the similarity of two fuzzy sets requires calculating the size of intersection and union of two membership functions. As triangular membership functions are used in our FNN, it makes such calculations simple and straightforward.

If an input variable ends up with only one membership function, which means that this input is irrelevant, we delete the input. We can thus eliminate irrelevant inputs and reduce the size of the rule base. If the classification accuracy of the FNN is below the requirement, and the number of rules is less than the specified maximum, we modify the rule base following the method introduced in next section.

2.5 Rule Base Modification

An additional membership function is added for each input at its value at the point of the maximum output error, following Higgins and Goodman[6]. One vertex of the additional membership function is placed at the value at the point of the maximum output error and has the membership value unity; the other two vertices lie at the centers of the two neighboring regions, respectively, and have membership values zero. As the output of the network is not a binary 0 or 1, but a continuous function in the range from 0 to 1, by firstly eliminating the error whose deviation from the target value is the greatest, we can speed up the convergence of the network substantially.

The rules generated above are then evaluated for accuracy and simplicity. We use a weighting parameter between accuracy and simplicity, which is the compatibility grade (*CG*) of each fuzzy rule. *CG* of rule j is calculated by the product operator as:

$$\mu_j(x) = \mu_{j1}(x_1) \times \mu_{j2}(x_{21}) \times \cdots \times \mu_{jn}(x_n) \quad \text{---- (9)}$$

when the system provides correct classification result.

All rules whose *CG* falls below a pre-defined threshold are deleted. Elimination of rule nodes is rule by rule, i.e., when a rule node is deleted, its associated input membership nodes and links are deleted as well. By varying the *CG* threshold the user is able to specify the degree of rule base compactness. The size of the rule base can thus be kept minimal. If the classification accuracy of the FNN after the elimination of rule nodes is below the prescribed requirement we will add another rule as described above, otherwise we stop the process.

Our FNN contains the powerful features of initial fuzzy model self-generation[16], fast input selection, partition validation [2, 6], parameter optimisation[18] and rule-base simplification, and combines them together to achieve better performance. Our fast inputs selection is added as an effective supplementation, which is much simpler than others' methods[14, 15].

3. EXPERIMENTAL EVALUATION FOR SYNTHESIZED DATASETS

To test the FNN we used ten classification problems of different complexity defined by R.Agrawal[1] on synthetic database with nine attributes. Attributes *elevel*, *car* and *zipcode* are categorical, and all others are non-categorical.

The first classification problem has predicates on the values of only one attribute. The 2nd to 3rd problems have predicates with two attributes, and the 4th to 6th problems have predicates with three attributes. Problems 7 to 9 are linear functions and problem 10 is a non-linear function of attribute values.

For comparisons with the FNN, we used decision tree construction algorithms C4.5 and C4.5rules[10] on the same data sets. C4.5 and C4.5rules Release 8 from the pruned trees with default parameters were used. We also compared our results with those of a pruned feedforward crisp neural network (NeuroRule)[8] for the classification problems.

Table 1 shows the accuracy on each test data set and the number of rules for all three approaches for the problems.

Compared to NeuroRule, the FNN produces rule bases of less complexity for all problems, except problem 7 and 9. The FNN gives better accuracy for problems 6, 7 and 9, and comparable but lower accuracy for the rest. Compared to C4.5rules, the FNN gives less complex rules for all problems. The FNN gives higher accuracy than C4.5rules on problems 6 to 10, and comparable but lower accuracy for the rest. The results for problems 8 and 10 were not reported in Lu's work[8] (indicated by "N/A" in Table 1).

Func	Accuracy (%)			Number of Rules		
	NR	C4.5	FNN	NR	C4.5	FNN
1	99.9	100	100	2	3	2
2	98.1	95.0	93.6	7	10	5
3	98.2	100	97.6	7	10	7
4	95.5	97.3	94.3	13	18	8
5	97.2	97.6	97.1	24	15	7
6	90.8	94.3	95.0	13	17	8
7	90.5	93.5	95.7	7	15	13
8	N/A	98.6	99.3	N/A	9	2
9	91.0	92.6	94.1	9	16	15
10	N/A	92.6	96.2	N/A	10	8

Table 1: Experimental Results 1

If very high accuracy is required, without consideration of compactness of rule base, we can set a lower threshold for rule elimination stage. From Table 2, it is found that we got higher accuracy compared with both methods in most of problems, except in problem 3 and 5. However, rule base becomes much larger than results in Table 1, which may make the analysis of database more difficult.

Func	Accuracy (%)			Number of Rules (%)		
	NR	C4.5	FNN	NR	C4.5	FNN
1	99.9	100	100	2	3	11
2	98.1	95.0	98.5	7	10	25
3	98.2	100	99.3	7	10	36
4	95.5	97.3	98.1	13	18	120
5	97.2	97.6	97.2	24	15	81
6	90.8	94.3	96.5	13	17	50
7	90.5	93.5	97.7	7	15	52
8	N/A	98.6	99.1	N/A	9	15
9	91.0	92.6	95.2	9	16	65
10	N/A	92.6	97.7	N/A	10	50

Table 2 : Experimental Results 2

4. EXPERIMENTAL EVALUATION FOR CANCER CLASSIFICTION

Microarray data analysis is one of those attractive fields of data mining. With the help of gene expression obtained from microarray technology, heterogeneous cancers can be classified into appropriate subtypes. Recently, different kinds of machine learning and statistical methods have been used to analyze gene expression data.

The small round blue cell tumors (SRBCT) data set[20] contains the expression data of 2308 genes. There are totally 63 training samples and 25 testing samples provided[20], 5 of the testing samples are not SRBCTs. The 63 training samples contain 23 Ewing family of tumors (EWS), 20 rhabdomyosarcoma (RMS), 12 neuroblastoma (NB) and 8 Burkitt lymphomas (BL). The 20 SRBCT testing samples contain 6 EWS, 5 RMS, 6 NB and 3 BL. In this data set, a small part of data is missing. A k-nearest neighbor algorithm was applied to fill those missing values[24].

In the first step, we ranked the entire 2308 genes according to their t-scores (TSs) [22,23] in the training data set. We picked out the 30 genes with the highest TSs. In this paper, each gene is labeled after its importance rank. For example, gene 5 means the gene ranked 5. Through its ID in the microarray (for example, GENE2046), the real name of each gene can be found on the website of the SRBCT data set[20].

Then we used the FNN to classify the SRBCT microarray data set. We found that the FNN performed very well: it can reach 100% accuracy for both the training data and the testing data with only the first 8 genes. Our result shows that the microarray data classification problem can be solved with a much smaller number of genes compared to previously published methods.

In Table 3, we made a comparison on the classification result of our FNN classifier with nearest shrunken centroid[21] and the ANN classifier proposed by Khan[20]. In this table, we found all the three classifiers can get 100% classification accuracy, but the number of genes used for classification shows great difference. Our FNN classifier use only 8 genes in sharp contrast with 96 genes used by ANN and 43 genes used in nearest shrunken centroid.

Method	Accuracy	Number of Gene
ANN	100%	96
Nearest Shrunken Centroid	100%	43
Fuzzy Neural Network	100%	8

Table 3: A Comparison of classification results of ANN, nearest shrunken centriod and FNN for SRBCT data set

The number of genes used to classify different cancers and the classification accuracy are the two key factors of crucial importance. In view of these two factors, we conclude that the FNN classifier not only helps biological researchers differentiate cancers that are difficult to be classified using traditional clinical methods, but also helps researchers focus on a small number of important genes to find the relationship between those important genes and the development of cancers.

5. DISCUSSION AND CONCLUSIONS

In this paper, we describe a novel Fuzzy Neural Network (FNN) and demonstrate its applications to data classification, especially for the problem of cancer classification based on microarray data.

In Table 1, we attempted to obtain the most compact rule base for the FNN to allow for easy analysis of the rules for very large databases, and subsequently easy decision making in real-world situations. If accuracy is more important than compactness of the rule base, it is possible to use the FNN with more strict accuracy requirements, i.e., a lower threshold for pruning the rule base, thereby producing more accurate results at the expense of a higher rule base complexity. The final decision regarding complexity versus accuracy of the rules is application specific.

Moreover, the performance of the FNN is significantly better for classes with relatively small amounts of available data, in comparison with C4.5rules, i.e., the FNN approach to data mining is insensitive to *the problem of small disjuncts*, in contrast to decision tree approaches such as C4.5rules[12]. This is due to the fact that the FNN treats all classes with equal importance, while decision trees give preference to more commonly occurred classes and treat classes with less data as less important.

The proposed FNN has also achieved outstanding results for cancer classification based on microarray data. Since the number of genes used to classify different cancers is a factor of crucial importance, smaller number of genes may help researchers directly focus their attention on some specific genes, which perhaps will lead to discovery of deep reasons of the development of cancers and discovery of drugs.

REFERENCES

1. R. Agrawal, T. Imielinski, and A. Swami, "Database Mining: A Performance Perspective," *IEEE Trans. Knowledge and Data Engineering*, vol. 5, no. 6, pp. 914-925, Dec. 1993.
2. C.T Chao, Y.J. Chen, and C.C. Teng, "Simplification of Fuzzy-Neural Systems Using Similarity Analysis," *IEEE Trans. Syst.,Man, Cybern, Part B: Cybernetics*, vol.26, no.2, pp.344-354, 1996.
3. S. E. Fahlman and C. Lebiere, "The cascade-correlation learning ar-chitecture," in *Advances in Neural Information Processing Systems II*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, pp. 524-532.
4. L. M. Fu, "Knowledge-based connectionism for revising domain theories," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, pp. 173-182, 1993.
5. C. L. Giles, D. Chen, G. Z. Sun, H. H. Chen, Y. C. Lee, and M. W. Goudreau, "Constructive learning of recurrent neural networks: Limitations of recurrent cascade correlation and a simple solution," *IEEE Trans. Neural Networks*, vol. 6, pp. 829-836, 1995.
6. C.M. Higgins and R.M. Goodman, "Fuzzy rule-based networks for control," *IEEE Transactions on Fuzzy Systems*, vol.2, no.1, pp. 82 -88, Feb. 1994.
7. J.R. Jang, "Input Selection for ANFIS Learning", *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, Vol.2, pp.1493-1499, 1996
8. H. Lu, R. Setiono, and H. Liu, "Effective Data Mining Using Neural Networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 957-961, Dec. 1996.
9. M. Mezard and J. P. Nadal, "Learning in feedforward layered networks: The tiling algorithm." *J. Phys.*, vol. 22, pp. 2191-2204, 1989.
10. J. R. Quinlan, *C4.5:Programs for Machine Learning*, Morgan Kaufmann: San Mateo, CA, 1993.
11. G. G. Towell and J. W. Shavlik, "Knowledge-based artificial neural networks," *Artif. Intell.*, vol. 70, pp. 119-165, 1994.
12. G. M. Weiss, "Learning with rare cases and small disjuncts," *Proceedings of the Twelfth International Conference on Machine Learning*, Morgan Kaufmann: San Francisco, CA, pp. 558-565, July 1995.
13. J.-S.R. Jang. "ANFIS: Adptive-Network-based Fuzzy Inference Systems." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 23, pp. 665-685, 1993.
14. C. Juang and C. Lin, "A recurrent self-organizing neural fuzzy inference network" *IEEE Trans. Neural Networks*, vol. 10, no.4, pp. 828-845, 1999.
15. Y. Frayman and L. Wang, "Data Mining Using Dynamically Constructed Fuzzy Neural Networks," *Research and Development in Knowledge Discovery and Data Mining: Proceedings of the Second Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-98)*, Springer-Verlag, Lecture Notes in Artificial Intelligence, vol. 1394, pp. 122-131, April 1998.
16. L. X. Wang and J. M. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 1414-1427, 1992.
17. M.Sugeno and G.T.Kang, " Structure identification of fuzzy model," *Fuzzy Sets Syst.*, vol. 28, pp. 15-33,1988.
18. P.Werbos, "Beyond regression:New tools for prediction and analysis in the behavioral sciences," Ph.D. dissertation, Harvard Univ.,Cambridge, MA, 1974.
19. D.Dubois and H.Prade, "A unifying view of comparision indices in a fuzzy set theoretic framework," *Fuzzy Sets and Possibility Theory: Recent Developments*, R.R.Yager,Ed. New York: Pergaon, 1982.
20. J. M. Khan, J. S. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, *et al.* Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks, *Nature Medicine*, Vol. 7, pp. 673-679, 2001.
21. R. Tibshirani, T. Hastie, B. Narashiman and G. Chu, Diagnosis of multiple cancer types by shrunken centroids of gene expression, *Proc. Natl. Acad. Sci. USA*, Vol. 99, pp. 6567-6572, 2002.
22. J. Devore, and R. Peck, *Statistics: the Exploration and Analysis of Data*, 3rd ed. Pacific Grove, CA.: Duxbury Press, 1997.
23. V. G. Tusher, R. Tibshirani, and G. Chu, Significance analysis of microarrays applied to the ionizing radiation response, *Proc. Natl. Acad. Sci. USA*, vol. 98, pp. 5116-5121, 2001
24. O. Troyanskaya, M. Cantor, G. Sherlock, et al., Missing value estimation methods for DNA microarrays, *Bioinformatics*, Vol. 17, pp. 520-525, 2001.