Pp. 775–784 in *Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence & Expert Systems (IEA-98-AIE)*, edited by A.P. del Pobil, J. Mira and M. Ali. Vol. 1416 in Lecture Notes in Artificial Intelligence, Springer-Verlag, 1998.

Comparing Soft Computing Methods in Prediction of Manufacturing data

Esa Koskimäki[1], Janne Göös[2],
Petri Kontkanen[3], Petri Myllymäki[3], and Henry Tirri[3]

[1] Helsinki University of Technology, Department of Information Technology
Automation Laboratory, P. O Box 3000, FIN-02015 HUT, Finland
[2] VTT Electronics, P. O. Box 1100, FIN-90571 Oulu, Finland
[3] Complex Systems Computation Group (CoSCo)
P.O.Box 26, Department of Computer Science
FIN-00014 University of Helsinki, Finland

Abstract. In the literature there exist several soft computing methods for building predictive models: neural network models, fuzzy models and probabilistic approaches. In this paper we are interested in the question which one of these approaches is likely to give best performance in practice. We study this problem empirically by selecting a set of typical models from the different model families, and by experimentally evaluating their predictive performance. For the evaluation, we use two real-world manufacturing datasets from a production plant of electrical machines. The models considered here include fuzzy rulebases, various neural network models and probabilistic finite mixtures. Our investigation indicates that all the methods can produce predictors that are accurate enough for practical purposes. Moreover, the results show that adding expert knowledge leads to improved predictive performance in the domain where such knowledge was available. In the domain where no expert knowledge was available, the probabilistic approach produced the best results.

1 Introduction

Real-world phenomena are complex, as indicated by increased interest in the "Science of complexity" [5]. Soft computing methods [31], such as neural networks [11] or fuzzy rules [10], address this complexity by providing methods for building models of the underlying complex systems. In this paper we are interested in soft computing approaches for building predictive models, where the task at hand is either to predict a continuous value (regression), or to select one action from a finite set of possible alternatives (classification). Our purpose is to compare different soft computing methods in real environments by testing how well they perform with real-life data. In this paper we have chosen as our evaluation framework two prediction problems in the manufacturing domain: estimation of manufacturing time for windings, which is a regression problem, and prediction of the required size for the synchronous machine to be built, which forms a classification task. The two datasets used for building the models come from a company producing synchronous machines. The study at hand is part of a larger project where soft computing methods are utilized in building sales support systems for synchronous machines [9, 16].

The different types of models considered in this paper include fuzzy rulebases, neural networks, and probabilistic models. We are currently not aware of any other comparison which includes all these three types of models. Moreover, in most similar predictive performance studies the data used had been artificially generated, but it is well-known that while artificial data may be good for initial validation of models, in practice the complexity of natural data makes the situation completely different. In our case the datasets chosen have practical importance to the company providing the data, and the prediction results also produce valuable information for the company.

In Section 2, we shortly discuss the two datasets used in the comparison. Section 3 presents the three model families considered in the study. Fuzzy rulebases are discussed in Section 3.1, and as the discussion in this section is not particularly deep, for further details we refer to [17, 10]. In Section 3.2 two different types of neural networks are considered: the standard multilayer perceptron [11], and the general regression neural network [24]. In Section 3.3, we present the family of probabilistic models used in this study, the family of finite mixtures [8, 28]. For the regression task we used the standard mixture of Gaussians model described in, e.g., [18]. For the latter problem, classification of the machine size, we used discrete finite mixtures with MAP prediction (see, e.g., [25]), as well as a computationally more efficient version called the Naive Bayes classifier with evidence prediction [15]. Section 4 reports the details of the models in the experiments (number of parameters, scaling etc.), and presents the empirical results for both of the data sets. We also briefly discuss some possible reasons for the observed performance of the different models. Section 5 gives the conclusions.

2 The problems and the data sets

The first problem studied in this paper concerns prediction of manufacturing time (in hours) of machine windings. This task forms an interesting problem since machine winding time has a direct relationship with the cost of building the machine, so it is an important factor in calculating the sales bid. However, the model has so many parameters that solving the problem by tables, which is a typical solution used for estimation at manufacturing plants, is not feasible, and hence estimation of the machine winding time has traditionally been considered a very difficult problem.

The second problem studied in this paper is a classification task, where the goal is to predict the size of the machine, given the customer inputs. The machine size is the most important factor in the design of a synchronous machine as the frame size determines many of the standard components needed to build the machine. In our case the machine size was discretized into 7 intervals, where each interval corresponded to a class value.

It should be observed that the above mentioned two problems are very different from the modeling point of view, as the first one is a regression problem while the latter requires building a classifier. The properties of the two datasets used in this study are summarized in Table 1. The winding data (Manufacturing hours) was collected from the document archives at the plant, and was originally not in machine-readable form. The machine size data (Diameter) was extracted from the plant database. The original database size was about 2000 records, but the data set was reduced to its reported size (349) by omitting the records with missing fields.

Table 1. Description of the datasets used in our experiments.

Dataset	*Size*	*#Attributes*	*#Classes*
Manufacturing hours	93	5	n/a
Diameter	349	9	7

3 The models

In the following we assume that we have m variables (continuous or discrete) $A_1 \dots A_m$, which are used to predict the value of an unknown variable C. We consider here both the case where C is continuous and where it is discrete. In the first dataset case (the the manufacturing hours data), C is continuous and we have a regression problem at hand. In the second dataset case (the Diameter data), C is discrete, and the goal is to build a classifier.

We consider here three different types of models: fuzzy rulebases (with expert knowledge), neural networks, and probabilistic models (in our case finite mixtures). It should be observed that not all of the models were tested with both datasets. For the latter problem, where no expert knowledge for constructing the fuzzy rules was available, the fuzzy approach with stochastic search turned out to be computationally too expensive. In this case more effort was used instead in testing a much wider variety of neural network models than with the first problem. It should be noted that although we used in the experiments a large collection of different neural network architectures, only the most successful structural variants are reported here. In the following, we describe the models used in more detail.

3.1 Fuzzy rulebase models

The first model family considered in the comparison consists of fuzzy rulebases. The fuzzy rulebases used here contain AND type rules where the MIN function was used both as the AND method and as the implication method, and the "center of gravity" method was used for defuzzification [30]. A single fuzzy IF-THEN rule then assumed the following form:

$$\text{IF } A_1 = a_1 \text{ AND } \dots \text{ AND } A_m = a_m \text{ THEN } C = c,$$

where $a_1, \dots, a_m, c$ are linguistic values defined by fuzzy sets. All the rules in the rule set had the same structure with the same attributes to be tested, i.e., all the condition parts of the rules contained the same set of attributes $A_1, \dots, A_m$, but with different values $a_1, \dots, a_m$.

The input membership functions were computed by using the trapezoidal and the two-sided Gaussian curve membership functions [17]. The shape of the Gaussian membership functions was controlled by four parameters: the midpoint, the width of the flat part and the width of both of the shoulders. The output membership functions used were triangle-type functions (for a good introduction to fuzzy concepts and principles, see [10, 17]).

The parameters of the rulebases were learned by using stochastic search algorithms [1, 2, 6]. In the literature there exist several studies where stochastic methods have been used for generating fuzzy rules from numerical data [12, 17]. In our experiments we used genetic algorithms and simulated annealing for this task. The parameters for the fuzzy rules were chosen so that the mean squared error in the training set was minimized.

Due to the reasons discussed already earlier, the fuzzy models were applied with the manufacturing hours data only. In this case, domain expertise from the factory was utilized. In particular, the experts defined the importance order of the input attributes affecting the manufacturing hours. Based on the importance of the attributes, the input space was partitioned into 2–4 membership functions. In general we followed the principle that the more important attributes were allowed more membership functions.

3.2 Neural network models

Two neural network types were included in our comparison: the standard multilayer perceptron (MLP) as described in, e.g., [11], and the general regression neural network (GRNN) [24]. Multilayer perceptrons are feedforward networks consisting of units that constitute the different layers of the network. The input propagates through the network in forward direction, and the networks are trained by supervised manner using error back-propagation algorithm [11]. The architecture and parameters of the multilayer perceptron were learned experimentally by minimizing the mean squared error in the validation set. Multilayer perceptrons were used with both datasets.

General regression neural networks are memory-based feedforward networks based on the estimation of probability density functions [24]. In GRNN models, the probability density functions are estimated by using Parzen windows as in probabilistic neural networks [23], but the the outputs of the units are weighted in proportion to the observed values of the dependent variable for the training patterns. In GRNN networks the only adjustable parameter, the smoothing factor for the kernel shape, is optimized by minimizing the validation set error during the training. In our tests, the GRNN network was trained by a "standard" genetic algorithm. In our experiments we used a version with binary chromosomes, elitist selection, population size 50, and cross-over probability 0.7. In our experiments the GRNN networks were used with the manufacturing hours data only.

3.3 Probabilistic models

The probabilistic model family chosen for the comparison was the family of *finite mixture models* [8, 28]. A finite mixture model for a set of random variables is a weighted sum of a relatively small number of independent mixing distributions (for a more formal description see, e.g., [25]). The main advantage of using finite mixture models lies in the fact that the computations for probabilistic reasoning can be implemented as a single pass computation. The model also conforms interestingly to the intuitively appealing *memory-based reasoning* paradigm (see e.g., [14]), and it can be seen to offer a Bayesian solution to the case matching and case adaptation problems in such domains [27, 26].

When compared to single parametric distribution methods and nonparametric methods (such as kernel density estimators [22]), finite mixture models have many appealing properties. Firstly, in natural domains the distribution to be modeled is often so complicated that probabilistic modeling through single parametric distribution (e.g., normal, multinomial, Poisson) does not lead to good approximations. It seems that distribution multimodality is more a rule than an exception, which naturally leads to serious problems for classical models. Finite mixtures have a natural means to model multimodality by placing a component distribution around each mode. Secondly, finite mixture models are universal in the sense that they can approximate any distribution arbitrarily close as long as a sufficient number of component densities is used. Thirdly, many non-parametric approaches depend on the sample size. So if the training sample is large, non-parametric techniques are inefficient in terms of computation time and space. Finite mixtures instead compress the information given by the sample at two levels: first by grouping similar data instances to a same cluster, and second by modeling a set of cluster instances by a single mixing distribution.

Finite mixture models are an ideal model family for massively parallel hardware [21], and thus can be realized by neural architectures [19, 20]. In continuous domains mixture models have proven to be a viable family for supervised learning tasks as demonstrated by the "mixtures of experts" architecture of Jordan et al. [13, 29] (see also the work by Bishop et al. on "mixture density networks" [3, 4]). In this study, the finite mixture models were tested with both datasets. In the regression task, the standard finite mixture of Gaussians [18, 4] with uniform priors was used, and the mean of the posterior predictive likelihood was then taken to be the predicted value. In the classification task, all continuous attributes were discretized, and the family of discrete finite mixtures was used. In this case, we also experimented with a simplified, computationally efficient version of finite mixtures, the Naive Bayes model with evidence prediction [15] and uniform priors.

4 Empirical results

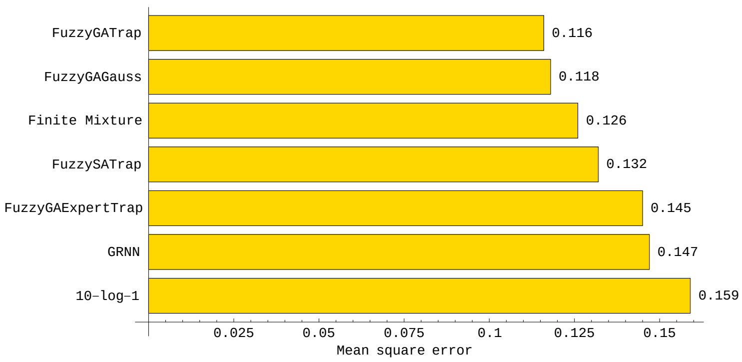
In the empirical studies the two datasets were tested with 10-fold crossvalidation. In the prediction of manufacturing hours, the prediction accuracy was measured using the mean squared error (MSE) measure. For the classification task we used the error rate, i.e., the ratio of incorrect classifications to size of the test set, as the performance measure.

4.1 The manufacturing hours data

In the first dataset case the goal was to predict the manufacturing time (in hours) of machine windings. There were 5 input attributes (see Table 1), of which the factory experts identified the first two being the most important ones. In the fuzzy rulebase models these attributes were assigned 4 and 3 membership functions, respectively. The other 3 attributes were assigned only 2 membership functions. The output space was partitioned into 8 triangular shape membership functions. These numbers of membership functions were chosen by the experts, and were used in all the fuzzy models.

The results of the 10-fold crossvalidation runs are presented in Figure 1. With this dataset, the "default" method, i.e., a dummy predictor which always predicts the mean value of the target variable in the training set, irrespective of the values of the other variables, yields a mean squared error of 0.257.

Fig. 1. The results with the manufacturing hours data.



In our experiments, the fuzzy rulebase always consisted of a fixed number of rules (15). The parameters were selected by repeating the stochastic search several times from different starting points. The model corresponding to the lowest MSE measure was then chosen. In the experiment 'FuzzyGATrap', the genetic algorithm was used for optimizing the rulebase, and the membership functions were trapezoidal. This combination was the one that yielded the best result. Similar rulebase with membership functions of the Gaussian type (FuzzyGA-Gauss) showed slightly worse performance, but was nevertheless better than a fuzzy rulebase with trapezoidal membership functions (FuzzySATrap) optimized by simulated annealing. In the final experiment the fuzzy rulebase used membership function shapes fixed by the experts (FuzzyGAExpertTrap), and only the rule parameters were generated by GA. However, the result suggests that in this case the expert knowledge could not be utilized effectively enough.

In the finite mixture experiment, the best result was achieved with 35 Gaussian components. For simplicity, all the components had the same (but unknown) variance. This is due to the well-known problem of continuous finite mixture maximum a posteriori (MAP) estimation [7]. Fixing this problem would probably lead to better performance. It should also be noted that the expert knowledge used for constructing the fuzzy rulebases could also probably be used for improving the performance of the standard mixture model used, but this idea was not tested in this set of experiments.

4.2 The diameter data

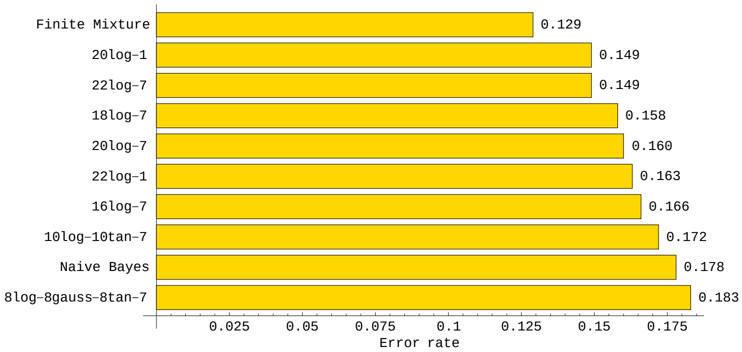
The second dataset used in our comparison was the diameter dataset. In this case, the goal was to predict the size of the machine frame, based on the customer specifications. The machine size was discretized into 7 intervals, and thus the problem can be regarded as a classification task.

In the case of the multi-layer perceptron, one hidden layer with variable number of neurons was used. All the signals in the network were scaled to the interval [0,1]. The shapes of the activation functions could be varied, but the output neurons had a logarithmic shape in all cases.

Network architectures with two types of output layer were tested: a layer with one output neuron, and a layer with one neuron for each class. In the latter case, the result of the classification of the network was interpreted to be the class corresponding to the neuron with the maximum output. In the case of one output neuron, the continuous output value was discretized into seven regions defining the classes. The size of the intervals representing the classes were chosen proportionally to the distance between the standard diameter measures, i.e., they were not uniform.

The error rates of the models tested can be found in Figure 2. The error rate of the "default" method, i.e., one minus the relative frequency of the most common value of the class variable in the training set, was in this case 0.602.

Fig. 2. The results with the the diameter data.



In these experiments, the discrete finite mixture model was the clear winner. The best results were obtained with 30–40 clusters. Between these limits, the actual choice for the number of clusters had little effect on the results.

The results suggest that the number of hidden units used in the multi-layer perceptron does not affect the results very much. Also, the two types of output layers (1-neuron and 7-neuron) seemed to produce quite similar performance. On the other hand, logarithmic hidden layer activation functions proved out to be clearly better than the Gaussian or the tangent ones.

It seems that the Naive Bayes classifier is probably a too restricted model for this particular task. One possible reason for this might be that the class attribute does not cluster the data well enough, and consequently the conditional independence assumption of the Naive Bayes model (attributes are independent given the class) fails. The model can, however, be useful in time-critical situations, since the Naive Bayes classifier is able to produce quite reasonable results within a fraction of a second.

5 Conclusion

We have discussed the use of different soft computing methods in prediction of manufacturing data. The wide spectrum of approaches considered makes the comparison particularly interesting. Although in the literature one is able to find many comparisons including fuzzy rulebases and neural network models, the addition of probabilistic models makes the study at hand unique. In the experimental part of the paper, the predictive performances of the chosen models were compared by using both a regression task and a classification task.

In the regression task, the observed differences between the models, in terms of the predictive accuracy, were negligible. The fuzzy rulebases with genetic algorithm parameter optimization, combined with expert knowledge in the form of attribute importance determination, yielded the best results. The additional expert knowledge, fixing the membership functions, did not improve the performance. There were no fuzzy rulebase tests without expert knowledge, so it is difficult to evaluate whether the success of fuzzy rulebases was due to the goodness of the model family or result from the expert knowledge. The Gaussian mixture family produced practically the same results than fuzzy rulebases, but the two neural network models did not perform very well. It should be noted that the results with the mixture models were obtained with no domain knowledge, although the same expert knowledge used for constructing the fuzzy rulebases could probably also be used for improving the performance of the standard mixture model used. This idea, however, could not be tested in this set of experiments.

In the classification task, finite mixtures produced clearly the best results. The choice of the number of clusters had little effect on the results. For the multilayer perceptron, the choice of the hidden layer size was also not too important, as long as a sufficient number of neurons (about 16) was used. Furthermore, the logarithmic activation function proved out to be clearly superior to the other candidates. The two output layer types considered, with one and seven neurons, were found to yield comparable results. The simple, computationally efficient version of mixture models, the Naive Bayes classifier, did not perform as well as the other approaches tested, which is somewhat surprising considering the extensive empirical tests in [25], where the Naive Bayes approach was found to perform consistently better than multi-layer perceptron (and many other algorithms). Nevertheless, it is well-known that there are some situations where the Naive Bayes classifier does not work well, for example in cases where there are only few classes in the data and the data is very disjunctive. This could provide a possible explanation for the results.

Finally, it should mentioned that the actual prediction results obtained in this study are very good with all the models tested. The outputs of the predictors are in practice very accurate, which indicates that the results of this study will be of practical relevance to the company in question.

Acknowledgements This research has been supported by the the Technology Development Center (TEKES), and the Academy of Finland.

References

1. E. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing.* John Wiley & Sons, Chichester, 1989.
2. J. Alander. An indexed bibliography of genetic algorithms. In L. Chambers, editor, *Practical Handbook of Genetic Algorithms: New Frontiers*, chapter Appendix 1, pages 333–427. CRC Press, Boca Raton, FL, 1995.
3. C. M. Bishop. Mixture density networks. Technical Report NCGR/4288, Neural Computing Research Group, Department of Computer Science, Aston University, 1994.
4. C.M. Bishop, M. Svensén, and C.K.I. Williams. EM optimization of latent-variable density models. In D.S. Touretzky, M.C.Mozer, and M.E.Hasselmo, editors, *Advances in Neural Information Processing Systems 8*. MIT Press, 1996.
5. J.L. Casti. *Complexification: Explaining a Paradoxical World through the Science of Surprise.* HarperCollins, New York, NY, 1994.
6. L. Davis. *Handbook of genetic algorithms.* Van Nostrand Reinhold, New York, NY, 1991.
7. R.O. Duda and P.E. Hart. *Pattern classification and scene analysis.* John Wiley, 1973.
8. B.S. Everitt and D.J. Hand. *Finite Mixture Distributions.* Chapman and Hall, London, 1981.
9. J. Göös and E. Koskimäki. Fuzzy fitness function for electric machine design by genetic algorithm. In J. Alander, editor, *Proceedings of the Second Nordic Workshop on Genetic Algorithms and their Applications*, pages 237–244, Vaasa, Finland, 1996.
10. C.J. Harris, C.G. Moore, and M. Brown. *Intelligent Control: Aspects of Fuzzy Logic and Neural Nets.* World Scientific, 1993.
11. S. Haykin. *Neural Networks: A Comprehensive Foundation.* IEEE Press/Macmillan College Publishing Company, New York, 1994.
12. H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka. Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on Fuzzy Systems*, 3(3):260–270, 1995.

13. M.I. Jordan and R.A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.
14. H. Kitano. Challenges of massive parallelism. In *Proc. of IJCAI-93, the Thirteenth International Joint Conference on Artificial Intelligence*, pages 813–834, Chambéry, France, August 1993. Morgan Kaufmann Publishers.
15. P. Kontkanen, P. Myllymäki, T. Silander, H. Tirri, and P. Grünwald. Comparing predictive inference methods for discrete domains. In *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, pages 311–318, Ft. Lauderdale, Florida, January 1997.
16. E. Koskimäki and J. Göös. Electric machine dimensioning by global optimization. In *Proceedings of the First International Conference on Knowledge-Based Intelligent Electronic Systems*, pages 308–312, Adelaide, Australia, 1997.
17. C.C. Lee. Fuzzy logic in control systems: fuzzy logic controller (part i and part ii). *IEEE transactions on systems, man and cybernetics*, 20(2):404–435, 1990.
18. D.V. Lindley. *Introduction to probability and statistics from a Bayesian viewpoint, Part 2 : Inference*. Cambridge University Press, 1965.
19. Jordan M.I. and C.M. Bishop. Neural networks. A.I.Memo 1562, MIT, Artificial Intelligence Laboratory, 1996.
20. P. Myllymäki and H. Tirri. Bayesian case-based reasoning with neural networks. In *Proceedings of the IEEE International Conference on Neural Networks*, volume 1, pages 422–427, San Francisco, March 1993. IEEE, Piscataway, NJ.
21. P. Myllymäki and H. Tirri. Massively parallel case-based reasoning with probabilistic similarity metrics. In S. Wess, K.-D. Althoff, and M Richter, editors, *Topics in Case-Based Reasoning*, volume 837 of *Lecture Notes in Artificial Intelligence*, pages 144–154. Springer-Verlag, 1994.
22. D.W. Scott. *Multivariate Density Estimation. Theory, Practice, and Visualization.* John Wiley & Sons, New York, 1992.
23. D. Specht. Probabilistic neural networks. *Neural Networks*, 3:109–118, 1990.
24. D.F. Specht. General regression neural network. *IEEE Transactions on Neural Networks*, 2(6):568–576, November 1991.
25. H. Tirri. *Plausible Prediction by Bayesian Inference.* PhD thesis, Report A-1995-1, Department of Computer Science, University of Helsinki, June 1997.
26. H. Tirri, P. Kontkanen, and P. Myllymäki. A Bayesian framework for case-based reasoning. In I. Smith and B. Faltings, editors, *Advances in Case-Based Reasoning*, volume 1168 of *Lecture Notes in Artificial Intelligence*, pages 413–427. Springer-Verlag, Berlin Heidelberg, November 1996.
27. H. Tirri, P. Kontkanen, and P. Myllymäki. Probabilistic instance-based learning. In L. Saitta, editor, *Machine Learning: Proceedings of the Thirteenth International Conference*, pages 507–515. Morgan Kaufmann Publishers, 1996.
28. D.M. Titterington, A.F.M. Smith, and U.E. Makov. *Statistical Analysis of Finite Mixture Distributions.* John Wiley & Sons, New York, 1985.
29. S. Waterhouse, D. MacKay, and T. Robinson. Bayesian methods for mixtures of experts. In D.S. Touretzky, M.C.Mozer, and M.E.Hasselmo, editors, *Advances in Neural Information Processing Systems 8.* MIT Press, 1996.
30. L.A. Zadeh. Possibility theory and soft data analysis. In L. Cobb and R. M. Thrall, editors, *Mathematical frontiers of the social and policy sciences*, pages 69–129. Westview Press, Boulder, CO, 1981.
31. L.A. Zadeh. Fuzzy logic, neural networks, and soft computing. *Comm. ACM.*, 37:77–84, 1994.