Generating Rules from Data by Fuzzy and Neuro–Fuzzy Methods

(Datenbasierte Regelgenerierung mittels Fuzzy– und Neuro–Fuzzy–Methoden)*

Frank Klawonn, Detlef Nauck, and Rudolf Kruse
Department of Computer Science
Technical University of Braunschweig
Bültenweg 74 / 75, D-38106 Braunschweig, Germany
Tel.: +49.531.391.3155, Fax: +49.531.391.5936, Email: nauck@ibr.cs.tu-bs.de

*This paper is going to appear in the Proceedings of the Third German GI–Workshop "Fuzzy–Neuro–Systeme'95" to be held in Darmstadt, Germany, November 15 – 17, 1995

Kurzfassung

In diesem Beitrag stellen wir einen Ansatz zur Neuro–Fuzzy–Klassifikation vor, der Fuzzy–Mengen und Fuzzy–Regeln auf der Grundlage von Beispieldaten erlernen kann. Die Fuzzy–Regeln, die bei diesem Ansatz erlernt werden, lassen sich zwar sehr gut interpretieren, jedoch ist ihre Klassifikationsleistung nicht so gut, wie die anspruchsvoller Fuzzy–Clusteringverfahren. Diese liefern jedoch meist Fuzzy–Mengen, die eine Interpretation der resultierenden Fuzzy–Regeln stark erschwert. Eine Kombination beider Ansätze kann diese Nachteile vermeiden.

Abstract

In this paper we present an approach to neuro–fuzzy classification that is able to learn fuzzy sets and fuzzy rules from data. The fuzzy rules that are created by this approach can be very well interpreted, however, they do not classify as good as the rules derived by sophisticated fuzzy clustering algorithms. They on the other hand supply usually unsatisfying fuzzy sets what makes it hard to interpret the rules. Combining both approaches can eliminate these disadvantages.

1 Introduction

In this paper we discuss how fuzzy rules [5] can be derived from a set of data by combining methods of fuzzy cluster analysis with a neuro–fuzzy classification approach. The goal is to derive fuzzy rules from a set of data that can be separated in different classes. The fuzzy rules

describing the data are of the form:
if x_1 is μ_1 **and** x_2 is μ_2 **and** ... **and** x_n is μ_n **then** the pattern $(x_1, x_2, \ldots, x_n)$ belongs to class i,
where $\mu_1, \ldots, \mu_n$ are fuzzy sets. The task of our NEFCLASS model is to discover these rules and to learn the shape of the membership functions. The problem of this approach is, that the rule learning capabilities are less sophisticated than in fuzzy clustering algorithms. On the other hand the learning procedure of fuzzy sets allows to apply constraints on the modification of the membership functions. Thus we can derive fuzzy rules that can be much better interpreted, as rules discovered by fuzzy clustering, because these approaches supply only less intuitive fuzzy sets.

We propose to combine both approaches. We can first derive an initial fuzzy rule base by clustering algorithms, and can then learn the shape of suitable membership functions with a neuro–fuzzy approach. We will first present NEFCLASS, an approach to neuro–fuzzy classification, and discuss then fuzzy clustering methods to derive good initial fuzzy rules for the neuro–fuzzy model.

2 Neuro–Fuzzy Classification

NEFCLASS means **NE**uro **F**uzzy **CLASS**ification, and is used to determine the correct class or category of a given input pattern. The patterns are vectors $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and a class C is a (crisp) subset of $\mathbb{R}^n$. We assume the intersection of two different classes to be empty. The pattern feature values are represented by fuzzy sets, and the classification is described by a set of linguistic rules. For each input feature x_i there are q_i fuzzy sets $\mu_1^{(i)}, \ldots, \mu_{q_i}^{(i)}$, and the rule base contains k fuzzy rules $R_1, \ldots, R_k$.

The rule base is an approximation of an (unknown) function $\varphi : \mathbb{R}^n \to \{0,1\}^m$ that represents the classification task where $\varphi(\mathbf{x}) = (c_1, \ldots, c_m)$ such that $c_i = 1$ and $c_j = 0$ $(j \in \{1, \ldots, m\}, j \neq i)$, i.e. $\mathbf{x}$ belongs to class C_i. Because of the mathematics involved the rule base actually does not approximate φ but the function $\varphi' : \mathbb{R}^n \to [0,1]^m$. We will get $\varphi(\mathbf{x})$ by $\varphi(\mathbf{x}) = \psi(\varphi'(\mathbf{x}))$, where ψ reflects the interpretation of the classification result obtained from a NEFCLASS system. In our case we will map the highest component of each vector $\mathbf{c}$ to 1 and its other components to 0, respectively.

The fuzzy sets and the linguistic rules which perform this approximation, and define the resulting NEFCLASS system, will be obtained from a set of examples by learning. The Fig. 1 shows a NEFCLASS system that classifies input patterns with two features into two distinct classes by using five linguistic rules.

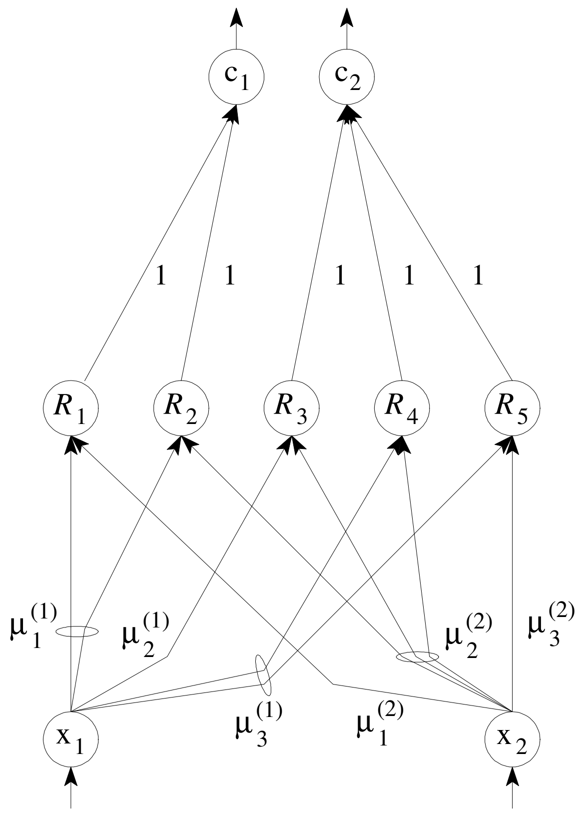

Figure 1: A NEFCLASS system with two inputs, five rules and two output classes

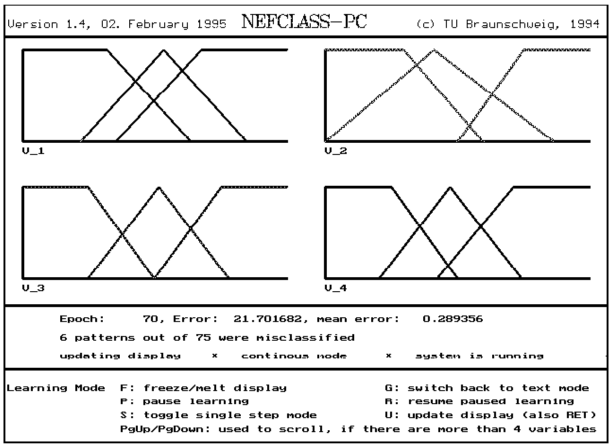

Figure 2: The graphical interface of the NEFCLASS software

Its main feature are the *shared weights* on some of the connections. This way we make sure, that for each linguistic value (e.g. "x_1 is *positive big*") there is only one representation as a fuzzy set (e.g. $\mu_1^{(1)}$ in Fig. 1), i.e. the linguistic value has only one interpretation for all rule units (e.g. R_1 and R_2 in Fig. 1). It cannot happen that two fuzzy sets that are identical at the beginning of the learning process develop differently, and so the semantics of the rule base encoded in the structure of the network is not affected [6]. Connections that share a weight always come from the same input unit. A NEFCLASS system is a special 3–layer fuzzy perceptron and can directly be derived from our model of a generic fuzzy perceptron [7]

A NEFCLASS system can be build from partial knowledge about the patterns, and can be then refined by learning, or it can be created from scratch by learning. A user has to define a number of initial fuzzy sets partitioning the domains of the input features, and must specify the largest number of rule nodes that may be created in the hidden layer.

A NEFCLASS system that is created from scratch starts with no hidden units at all. They are created during a first run through the learning task $\tilde{\mathcal{L}}$, i.e. the set of examples. A rule is created by finding for a given input pattern $\mathbf{p}$ the combination of fuzzy sets, where each yields the highest degree of membership for the respective input feature. If this combination is not identical to the antecedents of an already existing rule, a new rule node is created. After the data has been processed, the best k rules are selected, and all other rule nodes are deleted. The best rules are found by another run through the data set, and accumulating the individual rule error values weighted by the rule activation values.

After the rule base is created, the learning algorithm will adapt the membership functions of the antecedents. In this paper we consider usual triangular membership functions with three parameters a, b, c, where b defines the top, and a and c the left and right spreads. We also use min to determine the degree of fulfillment of a rule, i.e. the activation of a rule node.

Definition 1 (NEFCLASS learning algorithm)
Consider a NEFCLASS system with n input units $x_1, \ldots, x_n$, $k \leq k_{\max}$ rule units $R_1, \ldots, R_k$, and m output units $c_1, \ldots, c_m$. Also given is a learning task $\tilde{\mathcal{L}} = \{(\mathbf{p}_1, \mathbf{t}_1), \ldots, (\mathbf{p}_s, \mathbf{t}_s)\}$ of s patterns, each consisting of an input pattern $\mathbf{p} \in \mathbb{R}^n$, and a target pattern $\mathbf{t} \in \{0,1\}^m$. The learning algorithm that is used to create the k rule units of the NEFCLASS system consists of the following steps **(rule learning algorithm)***:*

(i) Select the next pattern $(\mathbf{p}, \mathbf{t})$ from $\tilde{\mathcal{L}}$

(ii) For each input unit $x_i \in U_1$ find the membership function $\mu_{j_i}^{(i)}$ such that $\mu_{j_i}^{(i)}(p_i) = \max_{j \in \{1,\ldots,q_i\}} \{\mu_j^{(i)}(x_i)\}$

(iii) If there is no rule node R with $W(x_1, R) = \mu_{j_1}^{(1)}, \ldots, W(x_n, R) = \mu_{j_n}^{(n)}$ than create such a node, and connect it to the output node c_l if $t_l = 1$.

(iv) If there are still unprocessed patterns in $\tilde{\mathcal{L}}$, and $k < k_{\max}$ then proceed with step (i); otherwise determine the best $k_{\max}$ rules, delete all other rule nodes, and stop.

The supervised learning algorithm of a NEFCLASS system to adapt its fuzzy sets runs cyclically through the learning task $\tilde{\mathcal{L}}$ by repeating the following steps until a given end criterion is met **(fuzzy set learning algorithm)***:*

(i) Select the next pattern $(\mathbf{p}, \mathbf{t})$ from $\tilde{\mathcal{L}}$, propagate it through the NEFCLASS system, and determine the output vector $\mathbf{c}$.

(ii) For each output unit c_i: Determine the delta value $\delta_{c_i} = t_i - o_{c_i}$

(iii) For each rule unit R with $o_R > 0$:

(a) Determine the delta value $\delta_R = o_R(1 - o_R) \sum_{c \in U_3} W(R, c)\delta_c$.

(b) Find x' such that $W(x', R)(o_{x'}) = \min_{x \in U_1} \{W(x, R)(o_x)\}$.

(c) *For the fuzzy set $W(x', R)$ determine the delta values for its parameters a, b, c using the learning rate $\sigma > 0$:*

$$\begin{aligned}
\delta_b &= \sigma \cdot \delta_R \cdot (c-a) \cdot \operatorname{sgn}(o_{x'} - b), \\
\delta_a &= -\sigma \cdot \delta_R \cdot (c-a) + \delta_b, \\
\delta_c &= \sigma \cdot \delta_R \cdot (c-a) + \delta_b,
\end{aligned}$$

and apply the changes to $W(x', R)$ if this does not violate against a given set of constraints Φ. (Note: the weight $W(x', R)$ might be shared by other connections, and in this case might be changed more than once)

(iv) *If an epoch was completed, and the end criterion is met, then stop; otherwise proceed with step (i).*

The rules found by this procedure can usually be very well interpreted due to the constraints Φ applied to the modifications of the membership function. However, the price of a sub–optimal classification has to be paid. It would therefore be a good idea to initialize the NEFCON model with rules derived from fuzzy clustering, and then apply the described learning procedure. The rule base can either be kept constant, or it can be allowed to create additional rules.

3 Fuzzy Cluster Analysis and Fuzzy Rules

Nearly all fuzzy clustering algorithms try to find an adequate prototype for each fuzzy cluster and suitable membership degrees for the data to each cluster. Usually, the cluster algorithm aims at minimizing the objective function

$$J(X, U, v) = \sum_{i=1}^{c} \sum_{k=1}^{s} (u_{ik})^m d^2(\mathbf{v}_i, \mathbf{p}_k) \tag{1}$$

under the constraints that each cluster must not be completely empty and usually that for every element the sum over its membership degrees to all clusters equals 1.

$X = \{\mathbf{p}_1, \ldots, \mathbf{p}_s\} \subseteq \mathbb{R}^n$ is the data set, c is the number of fuzzy clusters, $u_{ik} \in [0, 1]$ is the membership degree of datum $\mathbf{p}_k$ to cluster i, $\mathbf{v}_i \in \mathbb{R}^n$ is the prototype for cluster i, and $d(\mathbf{v}_i, \mathbf{p}_k)$ is the distance between prototype $\mathbf{v}_i$ and datum $\mathbf{p}_k$. The parameter $1 < m$ is called fuzziness index. For $m \to 1$ the clusters tend to be crisp, i.e. either $u_{ik} \to 1$ or $u_{ik} \to 0$, for $m \to \infty$ we have $u_{ik} \to 1/c$. Usually $m = 2$ is chosen.

The objective function (1) to be minimized uses the sum over the quadratic distances of the data to the prototypes weighted with their membership degrees.

The most simple fuzzy clustering algorithm is the fuzzy c–means (FCM) (see f.e. [1]) where the distance d is simply the Euclidean distance. It searches for spherical clusters of approximately the same size.

Gustafson and Kessel [3] and Gath and Geva [2] designed fuzzy clustering methods that are looking for hyper–ellipsoidal clusters of varying size. We refer to the corresponding algorithms by GK and GG, respectively, In both cases, in addition to the prototypes $\mathbf{v}_i$ and the membership degrees u_{ik} for each cluster i a (positive definite) covariance matrix C_i is calculated. Based on this matrix a transformed Euclidean distance is computed allowing for hyper–ellipsoidal clusters. In any case for each cluster a matrix inversion has to be computed in every iteration step so that these algorithms have a much higher computational complexity than the FCM.

For our purpose to generate rules from data by fuzzy cluster analysis the FCM is too restrictive, since it concentrates on spherical clusters of approximately the same size. As we will see, the GK and the GG cause trouble, since they are looking for clusters in the form of arbitrary hyper–ellipsoids, whereas for rule induction it would be more advantageous to have axes–parallel hyper–ellipsoids. This means that we have to restrict the positive definite matrices appearing in the GK and the GG to diagonal matrices. Since in the derivation of these algorithms the GG (fuzzy) covariance matrices are needed, one does in general not obtain a diagonal matrix. Therefore, we have derived new formulae for computing the matrices C_i that induce the transformation of the Euclidean distance, forcing it to be diagonal (see [4] for details). Neither the inverse nor the determinant of a matrix has to be computed in our algorithms so that they are much simpler and faster than the original GK and GG.

The principal idea of inducing classification rules based on fuzzy cluster analysis is the following. Each fuzzy cluster is assumed to be assigned to one class for classification. The membership grades of the data to the clusters determine the degree to which they can be classified as a member of the corresponding class. With a fuzzy cluster that is assigned to the class $\mathcal{C}$ we associate a linguistic classification rule in the following way. The fuzzy cluster i is projected into each single dimension leading to a fuzzy set on the real numbers. The correct, but computationally inefficient method of projecting a fuzzy cluster would lead to the fuzzy set

$$\mu_\nu(y) \;=\; \sup\left\{ \frac{1}{\displaystyle\sum_{j=1}^{c}\left(\frac{d^2(\mathbf{v}_i,\mathbf{x})}{d^2(\mathbf{v}_j,\mathbf{x})}\right)^{\frac{1}{m-1}}} \;\middle|\; x=(x_1,\ldots,x_{i-1},y,x_{i+1},\ldots,x_n)\in\mathbb{R}^n \right\}$$

as the νth projection of cluster i. Therefore, we use an approximation of this fuzzy set by projecting only the data set and computing the convex hull of this (discrete) projected fuzzy set or approximating it by a trapezoidal or triangular membership function as for instance proposed in [8].

To these fuzzy sets we assign suitable linguistic labels like *approximately zero* or *positive small* etc. (for example, one could assign the linguistic label approximately x_0 where x_0 is the value where the fuzzy set has its maximal membership degree).

The premise of the corresponding classification rule is the conjunction of these linguistic labels, the conclusion the class to which the cluster is assigned. The problem that arises is that the premise of the classification rule in the form of a conjunction leads to the Cartesian product of the corresponding one–dimensional fuzzy sets as the description of (a part of) the corresponding class. Unfortunately, the Cartesian product of projections is in general larger than the original fuzzy set (cluster) so that we have to accept a certain loss of information. As mentioned above, the FCM is very limited according to the restriction to spherical clusters of approximately the same size. On the other hand, the hyper–ellipsoidal clusters produced by the GK and the GG can result in misleading rules, since the projection of a hyper–ellipsoid may cause a strong loss of information. Therefore, we developed the modifications of the GK and GG in the previous section that are looking for hyper–ellipsoidal clusters with varying sizes whose axes are parallel to the coordinate axes. The advantage of these fuzzy clustering techniques is the greater flexibility compared to the FCM, the small loss of information when projected compared to the GK and GG and a simpler computation than for the GK and GG, since matrix inversion can be avoided.

Given a data set for which the correct classification is known, we start the fuzzy clustering algorithm with the number of clusters equal to the number of classes. To each cluster the class of the prototype or the class of the datum with the highest membership degree is assigned. We then determine for each cluster the rate of misclassifications. For each cluster in which the rate of misclassifications exceeds a given upper bound, a new prototype in the neighborhood of the original prototype of the cluster is introduced. After that we apply the fuzzy clustering algorithm again with increased number of clusters, i.e. the original number of clusters plus the number of newly introduced prototypes. As initialization the result of the clustering with the lower number of clusters incorporating the additional prototypes is used. We iterate this procedure until the number of misclassifications is small enough for each cluster.

In this way the number of rules is kept as small as possible. The classification capability of the resulting rules might be reduced by the approximation of the fuzzy sets that are induced by the projections of the clusters. Therefore it is advantageous to optimize these fuzzy sets by the NEFCLASS model which can use the rule base generated by the fuzzy clustering as initialization.

4 Conclusions

The combination of fuzzy clustering and neuro–fuzzy techniques for learning fuzzy classification rules seems to be very promising. NEFCLASS has proved itself to be very successful when it is initialized with suitable rule base in which only the fuzzy sets have to be optimized. On the other hand, fuzzy cluster analysis provides a very natural and adequate way of learning classification rules, whereas the fuzzy sets and their projections generated by fuzzy clustering do not always have a simple shape or lead to bad classification results according to the loss of information caused by the projection.

The NEFCLASS software (Fig. 2 shows its graphical surface) is freely available on the Internet at ftp.ibr.cs.tu-bs.de in the directory /pub/local/nefclass, or via the World Wide Web at http://www.cs.tu-bs.de/ibr/projects/nefcon/nefclass.html

References

[1] James C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, 1981.

[2] I. Gath and A.B. Geva. Unsupervised optimal fuzzy clustering. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 11:773–781, 1989.

[3] D.E. Gustafson and W.C. Kessel. Fuzzy clustering with a fuzzy covariance matrix. In *Proc. IEEE CDC*, pages 761–766, San Diego, 1979.

[4] F. Klawonn and R. Kruse. Constructing a fuzzy controller from data. *Fuzzy Sets and Systems*, 1995. To appear.

[5] Rudolf Kruse, Jörg Gebhardt, and Frank Klawonn. *Foundations of Fuzzy Systems.* Wiley, Chichester, 1994.

[6] Detlef Nauck, Frank Klawonn, and Rudolf Kruse. *Neuronale Netze und Fuzzy–Systeme.* Vieweg, Wiesbaden, 1994.

[7] Detlef Nauck and Rudolf Kruse. NEFCLASS – a neuro–fuzzy approach for the classification of data. In K.M. George, Janice H. Carrol, Ed Deaton, Dave Oppenheim, and Jim Hightower, editors, *Applied Computing 1995. Proc. of the 1995 ACM Symposium on Applied Computing, Nashville, Feb. 26–28*, pages 461–465. ACM Press, New York, February 1995.

[8] M. Sugeno and T. Yasukawa. A fuzzy–logic–based approach to qualitative modeling. *IEEE Trans. Fuzzy Systems*, 1:7–31, 1993.