Tatra Mt. Math. Publ. **16** (1999), 211–224

TATRA MOUNTAINS
Mathematical Publications

LEARNING STRATEGIES FOR NEURO-FUZZY CLASSIFICATION

GABRIELA ANDREJKOVÁ — HENRICH TÓTH

ABSTRACT. Neuro-fuzzy classification systems (NFCS) contain the fuzzy classification rules which are usually obtained by some algorithm. If the algorithm uses the data for the system construction then the algorithm is called a learning algorithm. D. Nauck and R. Kruse proposed some learning strategies for NFCS in [10, 12]. In this paper we discuss the learning algorithms for NFCS based on the modification of the fuzzy partitions defined on each single dimension and on the sequential improving of the knowledge base. The goal is to derive fuzzy rules from a data set that can be separated in different crisp classes. Our applications are concerned with using NFCS for the prediction of Geomagnetic Storms [2].

1. Introduction

An increasing has been observed interest in combining artificial neural networks and fuzzy systems. Various approaches to fuzzification of the neural networks have been proposed. Some of them use fuzzy logic, implementing IF–THEN rules, they are *neural fuzzy logic networks*; others use arithmetic fuzzy numbers, being *neural fuzzy algebraic networks*. A recent class of neuro-fuzzy systems is based on neural network topologies and on fuzzy logic systems with new learning strategies. Various learning strategies for neuro-fuzzy systems are discussed in [1, 9, 14]. A fuzzy classifier is not a replacement for the statistical methods or machine learning methods but it is a different way of achieving the same goal. Fuzzy systems have proven very successful in a variety of application [2, 3, 6, 9]. The effectiveness of the fuzzy models representing nonlinear input-output relationships depends on the fuzzy partition of the input-output space.

Section 2 of this paper contains the basic definitions. In Section 3, we recall some properties of a fuzzy system and neuro-fuzzy classifier. In Section 4, we

2000 Mathematics Subject Classification: 03E72, 68T05.
Keywords: computational learning, fuzzy sets, neuro-fuzzy classification.

This research was partially supported by Slovak Grant Agency for Science VEGA, project No. 1/4375/97.

describe the algorithm for the NFC construction (rule learning algorithm) and its learning algorithm (fuzzy set learning algorithm). In Section 5, we describe an application of FNC approach in a prediction of geomagnetic storms (GMS). Section 6 contains some concluding remarks.

2. Basic definitions

A fuzzy set A over some universe of discourse U (which is itself a classical set) is characterized by its membership function $\mu_A\colon U \to \langle 0,1\rangle$. A will be identified with this membership function.

The basic binary predicate symbol that denotes the membership relation of elements of the universe of discourse U with respect to some fuzzy subset of U will be "*is*" and the atomic formulas built up with it are of type "x is A". Their truth values correspond to the values of membership function (*membership degrees*) of the fuzzy sets such that for each fuzzy set A and each $x \in U$ the following fundamental equation is satisfied $[\![x \ is \ A]\!] \stackrel{\text{def}}{=} \mu_A(x)$.

The Cartesian product of two fuzzy sets A_1 and A_2 is defined [4] as $A_1 \times_t A_2 \stackrel{\text{def}}{=} \{(x_1,x_2)\colon x_1 \text{ is } A_1 \wedge_t x_2 \text{ is } A_2\}$, where t is t-norm.

The truth value of an expression "x_1 is A_1 AND x_2 is A_2" is computed as the membership degree of (x_1,x_2) in the fuzzy set $A_1 \times_t A_2$. In the paper, we use the set of real numbers $\mathbb{R}$ as the universe of discourse and Gödel's norm as the t-norm.

The rule has the following form: **IF** x is A **THEN** y is S,
where x, y are variables and A, S are fuzzy sets. The rule represents the fuzzy relation between the fuzzy sets A and S and it is a fuzzy subset of the Cartesian product $A \times_{t_G} S$, where t_G is the Gödel t-norm. Since the classical set is also a fuzzy set it is possible in the rule to use the classical set instead of S.

A space of examples X is the finite set of training examples.

A partition of X is a finite system of sets $C^1,\dots,C^m$, $m \geq 1$ such that $X = \cup_{i=1}^m C^i$ and $C^i \cap C^j = \emptyset$, for all $i \neq j$, $1 \leq i,\ j \leq m$. An element of the partition is called *a class. A classification problem* consists of finding a certain partition of X with some properties.

A triangular fuzzy set A is the fuzzy set over the universe of discourse $\mathbb{R}$ characterized by a triple of real numbers (l,c,r) with the following membership function:

$$\mu_A(x) = \begin{cases} \frac{x-1}{c-l} & \text{for } l < x \leq c\,, \\ \frac{r-x}{r-c} & \text{for } c < x < r\,, \\ 0 & \text{otherwise}\,, \end{cases}$$

where $l < c < r$, $l,c,r \in \mathbb{R}$.

3. Neuro-fuzzy classifiers

From our point of view, *neuro-fuzzy* means the employment of learning strategies derived from the domain of neural network theory to support the development of a fuzzy systems. The learning capability of neural networks made them [5, 7] a prime target for combination with fuzzy systems in order to automate or support the process of developing a fuzzy system. Modern neuro-fuzzy systems (NFS) are usually represented as multilayer feedforward systems but it is possible to use other network architecture and this means that the systems can have different properties. We consider the following properties of neuro-fuzzy systems [11]:

- NFS is a fuzzy system trained by a learning algorithm (usually derived from neural network theory).
- NFS can be described as a special feedforward NN. The units of NN use *t-norm* and *t-conorm* [4] instead of the activation function.
- NFS can always be interpreted as a system of fuzzy rules. The form of the rule depends on a real system. Very often the form is:
  IF x_1 is A_1 AND x_2 is A_2 AND ... AND x_n is A_n **THEN** y is S, where $x_1, \ldots, x_n, y$ are variables and $A_1, A_2, \ldots, A_n, S$ are fuzzy sets of values for linguistic variables.
- NFS approximates an n-dimensional (unknown) function that is partially given by the training data. The fuzzy rules encoded within the system represent vague samples, and can be viewed as vague prototypes of the training data.

The neuro-fuzzy classifier (NFC) is the modified NFS using the properties of neural networks. It is constructed to find a solution of the classification problem. The main difference between the NFS and NFC is in the structure of IF–THEN rules. The rule $R = (A_1, A_2, \ldots, A_n, C)$ for NFC has the following form:

$$\textbf{IF} \;\; x_1 \text{ is } A_1 \;\text{ AND} \ldots \text{AND }\; x_n \text{ is } A_n \;\; \textbf{THEN} \;\; \mathbf{x} \text{ belongs to } C\,,$$

where $\mathbf{x}$ is an example determined by actual parameters (attributes) $x_1, \ldots, x_n$. $A_1, \ldots, A_n$ are fuzzy sets and C is a class to which the example $\mathbf{x}$ belongs. The antecedent part of the rule will be designed by n-tuple $(A_1, \ldots, A_n)$.

Definition 1. A NFC system is a 4-layer (U_1, U_2, U_3, U_4) feed-forward neural network (where the first layer U_1 contains the input units representing the pattern feature, each unit in U_2 represents a fuzzy set, the third layer U_3 holds rule units representing the fuzzy rules and the fourth layer U_4 consists of output units, one for each class) with the following specifications:

1. $U_1 = \{x_1, x_2, \ldots, x_n\}$, $U_2 = \{A_j^i : i = 1, \ldots, n, j = 1, \ldots, q_i\}$, $U_3 = \{R^1, R^2, \ldots, R^k\}$ and $U_4 = \{c_1, c_2, \ldots, c_m\}$.

2. Each unit $A^i_j \in U_2$ is labelled with a linguistic term Term^i_j, $i = 1, \dots, n$, $j = 1, \dots, q_i$.
3. The weight values are:
   - between the first and the second layer $W(x_i, A^l_j) = 1$ if $i = l$, $W(x_i, A^l_j) = 0$ if $i \neq l$, for all $x_i \in U_1$, $i = 1, \dots, n$, $A^l_j \in U_2$, $j = 1, \dots, q_l$, $l = 1, \dots, n$,
   - between the second and the third layer are $W(A^i_j, R^r) \in \{0, 1\}$, for all $R^r \in U_3$, $r = 1, \dots, k$, A^i_j, $j = 1, \dots, q_i$, $i = 1, \dots, n$, and
   - between the third and the fourth layer $W(R^r, c_s) \in \{0, 1\}$ for all $R^r \in U_3$, $r = 1, 2, \dots, k$, $c_s \in U_4$, $s = 1, 2, \dots, m$.
4. For all rule units $R \in U_3$ and for all units $A^i_j, A^i_k \in U_2$ we have:

$$\big(W(A^i_j, R) = 1 \text{ and } W(A^i_k, R) = 1\big) \Longrightarrow k = j\,.$$

5. For all $R^{r_1}, R^{r_2} \in U_3$ we have:

$$\big[(\forall A^i_j \in U_2)\ W(A^i_j, R^{r_1}) = W(A^i_j, R^{r_2})\big] \Longrightarrow r_1 = r_2\,.$$

6. For all rule units $R \in U_3$ and all units $c, c' \in U_4$ we have

$$\big[W(R, c) = 1 \text{ and } W(R, c') = 1\big] \Longrightarrow c = c'\,.$$

7. Outputs o of neurons in single layers can be expressed as follows:
   - layer U_1: $o_{x_i} = x_i$;
   - layer U_2: $o_{A^i_j} = \mu_{A^i_j}(o_{x_i})$;
   - layer U_3: $o_{R^r} = \min\big\{o_{A^i_j} : W(A^i_j, R^r) = 1\big\}$
   - layer U_4: $o_c = \frac{\Sigma_{R^r \in U_3} W(R^r, c) \cdot o_{R^r}}{\Sigma_{R^r \in U_3} W(R^r, c)}$.
8. The outputs of NFC is m-dimensional vector $\mathbf{e}^i \in \{0, 1\}^m$ such that $\mathbf{e}^i = (0, \dots, 0, 1, 0, \dots, 0)$ with one 1 only in ith position. The index i is computed by $o_i = \max\limits_{c=1,\dots,m} \{o_c\}$.

Figure 1 shows an example of the neuro-fuzzy classifier for the knowledge base

R^1: IF x_1 is A^1_1 AND x_2 is A^2_1 THEN (x_1, x_2) belongs to C^1,
R^2: IF x_1 is A^1_2 AND x_2 is A^2_1 THEN (x_1, x_2) belongs to C^2,
R^3: IF x_1 is A^1_3 AND x_2 is A^2_2 THEN (x_1, x_2) belongs to C^1.

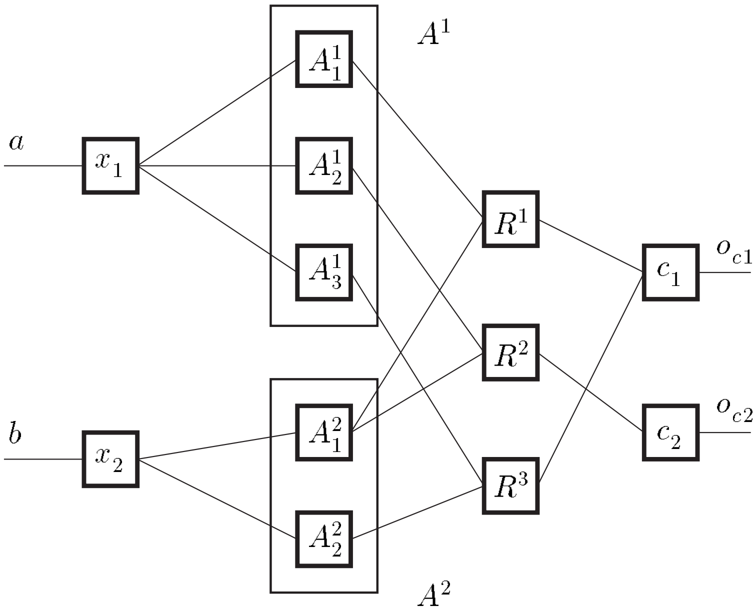

FIGURE 1. An example of the neuro-fuzzy classifier

The next significant difference between NFS and NFC is in the outputs of both networks. The output of NFC is an m-dimensional vector $\boldsymbol{e}^i \in \{0,1\}^m$ $i \in \{1, \ldots, m\}$. The output $\boldsymbol{e}^i$ determines the class C^i to which the example $\boldsymbol{x}$ belongs. The success of an NFC learning, is similar to that of an NFS, depends on the structure of the network and on the values of the parameters.

4. Learning of neuro-fuzzy classifiers from the data

Classifiers are usually derived from data and they are not directly specified. In the case of fuzzy classifiers two common methods are known [10]: *fuzzy clustering*, and *neuro-fuzzy learning*. In the case of the fuzzy clustering, the input space (input data) is searched for clusters of data and fuzzy rules are obtained by the projection of clusters. In the second case, the neuro-fuzzy classifier is obtained by a neuro-fuzzy learning procedure which will be analyzed in this paper.

Let T be a training sample of examples $(\boldsymbol{x}, \boldsymbol{c})$ with n parameters, that is $\boldsymbol{x} = (x_1, \ldots, x_n)$, $x_i \in X_i \subset \mathbb{R}$, $i = 1, \ldots, n$. Let $\mathcal{M}_i$ represent a fuzzy set system defined on X_i, for $i = 1, \ldots, n$, and let $C^1, \ldots C^m$ be classes. The algorithm constructs the system $\mathcal{B}$ of rules $R^r = \left(A_1^r, \ldots, A_n^r, C^{l_r}\right)$, $r = 1, \ldots, k$ of the following type:

$$R^r\colon \text{ IF } x_1 \text{ is } A_1^r \text{ AND } \dots \text{ AND } x_n \text{ is } A_n^r \text{ THEN } \boldsymbol{x} \in C^{l_r},$$

where A_i^r is a fuzzy set in $\mathcal{M}_i$, $i = 1, \dots, n$.

Each example of the training sample has to be classified according to the rules into the right class C^{l_r}, $l_r \in \{1, \dots, m\}$.

The algorithm starts from some initial knowledge base of fuzzy rules. The principle of the rule construction is as follows:

The rule R will be constructed for the example $(\boldsymbol{x}, \boldsymbol{c})$, $\boldsymbol{c} = \boldsymbol{e}^l$.

Let N_i be the fuzzy set $N_i \in \mathcal{M}_i$ that has the maximal membership value $\mu_{N_i}(x_i)$ of the element x_i in N_i among all the membership values of the element x_i in fuzzy sets in $\mathcal{M}_i$. N_i will be assigned to A_i. The class C^l corresponding to the vector $\boldsymbol{c} = \boldsymbol{e}^l$ will be assigned to C. The rule R will have the following form:

$$R\colon \text{ IF } x_1 \text{ is } N_1 \text{ AND } \dots \text{ AND } x_n \text{ is } N_n \text{ THEN } \boldsymbol{x} \in C^l, \tag{1}$$

where $\mu_{N_i}(x_i) = \max\limits_{M_i \in \mathcal{M}_i} \{\mu_{M_i}(x_i)\}$.

It is possible that the system of rules created contains some controversial rules that have the same antecedents but different consequences. The algorithm for the construction of the knowledge base has to create

(1) a system without controversial rules and

(2) a database of fuzzy sets as an input to the fuzzy set learning algorithm.

From the beginning, the spaces of parameters are partitioned into few (not many) fuzzy sets and the system of rules is constructed and optimized. If the created system is not appropriate then each fuzzy set A_i from the antecedent part of some controversial rule will be replaced by a few new fuzzy sets A_{i_r}, $r = 1, \dots, b_i$; $b_i \geq 2$ and a new system is prepared. The process of fuzzy set replacement and the optimization of the rule system (tuning, merging and deleting of the rules) is repeated until the system is appropriate or the allowed number of learning repetitions is overstepped.

Let us mention the deleting of rules from the system. It is only one of the heuristic methods to reduce the number of the rules in the system. But it is possible that it decreases the accuracy of the system. We use the deleting of rules for two reasons

1. a newly created rule specifies more an existing rule in the system (the inner structure of the existing rule will not be analyzed),
2. the tuning of the rules modifies the rules in such a way that there are two rules R^1, R^2 with the same consequent part and the support of the antecedent part of R^1 is a subset of the support of the antecedent part of R^2 (the rule R^1 is deleted to reduce the number of the rules in the system).

In the algorithm the symbols used have the following meaning:

P_l^A ... the number of patterns belonging to class C_l and leading to the construction of the antecedent part A according to (1),

F_l^A ... the sum of the firing levels of the antecedent part created according to the patterns above mentioned, where the firing level of the antecedent part $(A_1, \ldots, A_n)$ is $\min\limits_{1 \le i \le n} \{\mu_{A_i}(x_i)\}$,

$\mathrm{FL}(A, \boldsymbol{x})$ ... firing level of the antecedent part A for the pattern $\boldsymbol{x}$,

$\mathrm{ANT}(R)$ ... the n-tuple of fuzzy sets that forms the antecedent part of the rule R,

$\mathrm{CON}(R)$ ... the consequent part of the rule R,

left neighbour ... the fuzzy set $A = (l^A, c^A, r^A)$ is the left neighbour of the fuzzy set $B = (l^B, c^B, r^B)$ iff (i) $\operatorname{Supp} A \neq \operatorname{Supp} B$, (ii) $r^A \in \operatorname{Supp} B$ and $l^B \in \operatorname{Supp} A$,

$\mathrm{ANT}(R) \triangleright$ (*derives*) $\mathrm{ANT}(S)$ ... iff $\operatorname{Supp} B_i \subseteq \operatorname{Supp} A_i$, $i = 1, \ldots, n$, $R = (A_1, \ldots, A_n, C_R)$, $S(B_1, \ldots, B_n, C_S)$.

Rule learning algorithm.

The algorithm creates the rules for the training data and *the structure of the fuzzy neural network*. The algorithm can be used for an initialization of the network. We have found the motivation for the algorithm in [8, 10] and it has been modified. The algorithm is developed for the triangular shape of fuzzy sets but it can be modified for the other shapes of fuzzy sets.

Step 1. (*Initialization*)

Let d be the number of learning repetitions: $d := 1$;

Let $\mathcal{D} = \cup_{i=1}^{n} \mathcal{D}_i$ be the database of fuzzy sets. $\mathcal{D}_i$ be the system of fuzzy sets corresponding to the ith attribute.

$\mathcal{D} := \emptyset$;

Let $\mathcal{B}$ be the rule basis constructed by the database of fuzzy sets $\mathcal{D}$: $\mathcal{B} := \emptyset$;

For each $i = 1, 2, \ldots, n$ construct a system $\mathcal{M}_i$ of m_i fuzzy sets $B_{i_1}, B_{i_2}, \ldots, B_{i_{m_i}}$ as an initial full partitioning of the space X_i.

The above setting of system is applied if no prior knowledge exists for classification. In the other case, the existing information can be inserted into $\mathcal{B}, D$. The initial partition also has to be performed according to that information too.

Step 2. (*Antecedent constructions*)

$\mathcal{R} := \emptyset$, $\mathcal{A} := \emptyset$.

For each pattern $(\boldsymbol{x}, \boldsymbol{c})$, $\boldsymbol{c} = \boldsymbol{e}^l$ determine the antecedent part $A = (A_1, \ldots, A_n)$ according to (1).

If exists $R \in \mathcal{B}$ such that $\mathrm{ANT}(R) \triangleright A$ then go to the next pattern

(ANT(R) satisfies condition $\mathcal{P}$ and the next analysis of the rule R is stopped)
otherwise
if $A \notin \mathcal{A}$ then
insert A into $\mathcal{A}$ and set
$P_s^A := c_s$,
$F_s^A := c_s \cdot \mathrm{FL}(A, \boldsymbol{x})$,
for $s = 1, \dots, m$.

otherwise update
$P_s^A := P_s^A + c_s$,
$F_s^A := F_s^A + c_s \cdot \mathrm{FL}(A, \boldsymbol{x})$,
for $s = 1, \dots, m$.

Step 3. (*Selection rule phase*)
If $\mathcal{A} = \emptyset$ then the algorithm stops.
For each $A \in \mathcal{A}$ which satisfies condition, $\mathcal{P}$ determine its consequent part C_v: $F_v^A = \max\{F_s^A : s = 1, \dots, m\}$, create the rule $R = (A, C_v)$ and insert it into $\mathcal{R}$.
If $d = \mathrm{maxd}$ then create the rules $R = (A, C)$ for each $A \in \mathcal{A}$ and insert the rules into $\mathcal{R}$. maxd is the maximal number of learning repetitions.

Step 4. If $\mathcal{R} = \emptyset$ then go to step 8 else proceed with step 5.

Step 5. (*Adaptation phase*)
Construct the neural network for the rule base $\mathcal{R}$ according to Definition 1 and apply the fuzzy set learning algorithm.

Step 6. (*Merge phase*)
For each pair of rules $R^1, R^2 \in \mathcal{R}$ such that $\mathrm{ANT}(R^1) = (A_1^1, \dots, A_n^1)$, $\mathrm{ANT}(R^2) = (A_1^2, \dots, A_n^2)$ and $\mathrm{CON}(R^1) = \mathrm{CON}(R^2)$ do:

- If $\mathrm{ANT}(R^1) \triangleright \mathrm{ANT}(R^2)$ then $\mathcal{R} := \mathcal{R} - \{R^2\}$ (R^2 is removed from the rule base $\mathcal{R}$).
- If there exists $j \in \{1, \dots, n\}$ such that $A_i^1 = A_i^2$, $i \neq j$ and $A_j^1 = (l_j^1, c_j^1, r_j^1)$ is a *left neighbour* of $A_j^2 = (l_j^2, c_j^2, r_j^2)$ then a new rule R is created:
$R = (A_1^1, \dots, A_{j-1}^1, B, A_{j+1}^1, \dots, A_n^1, \mathrm{CON}(R^1))$,
where B is the new fuzzy set defined in the following way: $B = (l_j^1, x, r_j^2)$ and $x = \frac{c_j^1 + c_j^2}{2}$. The rule basis is modified in the following way: $\mathcal{R} := \mathcal{R} - \{R^1, R^2\} \cup \{R\}$.

Step 7. (*Moving optimized rules into the knowledge base*)
For each rule $R \in \mathcal{R}$ with $\mathrm{ANT}(R) = (A_1, \dots, A_n)$, $\mathcal{D}_i := \mathcal{D}_i \cup \{A_i\}$.

$\mathcal{B} := \mathcal{B} \cup \mathcal{R}$.
If $d = \mathrm{maxd}$ then algorithm stops.

Step 8. (*Partition phase*)
For each $i \in \{1, \ldots, n\}$ do:

- sort training set T with respect to increasing ith coordinate x_i,
- for each fuzzy set $B \in \mathcal{M}_i$,

– select T', $T' \subset T$ such that $\mathbf{x} = (x_1, \ldots, x_n) \in T'$ iff $x_i \in \operatorname{Supp} B$ and there does not exist the rule R, $R \in \mathcal{B}$ such that $\mathrm{ANT}(R) = A$, where A is the antecedent part determined by the pattern $\mathbf{x}$ according to (1).

– construct the system $\mathcal{S}^B$ of intervals $\mathcal{S}^B = \{(b_s^B, e_s^B)\}$ with bounds determined by real examples of T' (ith coordinate of examples). Each interval is the interval of maximal length and it has the following two properties: (1) it can not be shorter than ρ (the value ρ depends on the fuzzy set B) and (2) for each pair of the neighbour examples (after the sorting) $\mathbf{x}^k, \mathbf{x}^{k+1} \in T'$ holds $x_i^{k+1} - x_i^k < \delta$,

– if the previous construction does not create any interval ($\mathcal{S}^B = \emptyset$) then construct the system $\mathcal{S}^B$ of intervals by the uniform partition of $\operatorname{Supp} B$,

– construct fuzzy sets for created system of intervals and replace $B \in \mathcal{M}_i$ by the constructed fuzzy sets.

Step 9. $d := d + 1$; go to Step 2.

The success of the algorithm depends on a suitable choice of parameters:

– the maximal number of learning repetitions,

– the number of fuzzy sets for initial partitioning of the space X_i, $i = 1, \ldots, n$,

– the end criterion for the fuzzy set learning algorithm,

– of the new fuzzy sets for the replacement of one fuzzy set from the old system.

These parameters really influence the structure of the space partitions X_i by fuzzy sets from $\mathcal{M}_i$. The following dependence is a condition $\mathcal{P}$ for a (suitable) antecedent part $A \in \mathcal{A}$ that can be expressed as:

A is appropriate, if it satisfies at least one of next two inequalities:

$(\exists\, v)\ v \in \{1, \ldots, m\}\ (\forall\, s)\ s \in \{1, \ldots, m\}\ s \neq v,\ F_s^A \neq 0$

$$(1)\quad \frac{F_v^A}{F_s^A} \geq \alpha, \text{ resp. } F_v^A \geq \sum_{s=1,\ s\neq v}^{m} F_s^A,$$

(2) $\sum_{s=1}^{m} P_s^A \leq \beta$.

This means that *the appropriate antecedent part* satisfies the condition $\mathcal{P}$ if it is used in at least β examples of the training sample or there exists only one *dominant classification class* for it.

Fuzzy set learning algorithm.

This algorithm is constructed for the adaption of fuzzy sets in the system modelled above. The motivation can be found in [11]. In that algorithm, there are used properties of neural networks, especially their possibility of the learning from examples. It runs by repeating the following steps:

Step 1. Select the next pattern $(\boldsymbol{x}, \boldsymbol{c}) \in T$ and propagate it through the system and compute the output values o_{c_i}, $i = 1, \ldots, m$.

Step 2. For each output unit: Determine the error of the expected and the computed value, $\delta_{c_i} = c_i - o_{c_i}$.

Step 3. For each rule R with $o_R > 0$:

- $\delta_R = o_R(1 - o_R) \sum_{c \in U_4} W(R, c)\delta_c$,
- Find $A_{\mathrm{jmin}}^{\mathrm{imin}} \in U_2$ such that $o_{A_{\mathrm{jmin}}^{\mathrm{imin}}} = \min_{i,j} \{o_{A_j^i} : W(A_j^i, R) = 1\}$
- For the fuzzy set of the unit $A_{\mathrm{jmin}}^{\mathrm{imin}}$ determine the delta values for its parameters l, c, r using the learning rate $\sigma > 0$:
  $\delta_c = \sigma \cdot \delta_R \cdot (r - l) \cdot \mathrm{sgn}(o_{x_{\mathrm{imin}}} - c)$, $\delta_l = -\sigma \cdot \delta_R \cdot (r - l) + \delta_c$,
  $\delta_r = \sigma \cdot \delta_R \cdot (r - l) + \delta_c$,
  and apply the changes to the output function of the unit $A_{\mathrm{jmin}}^{\mathrm{imin}}$ if this unit has not been changed during the learning cycle.

Step 4. If an epoch was completed, and the end criterion is met, then algorithm stops, otherwise proceed by Step 1.

5. Application of the algorithms in the prediction of GMS

The application shows the Fuzzy Neural Network (FNN) approach, which to our knowledge has not yet been frequently used in Geomagnetic Storm (GMS) prediction. A neural network modelling of the prediction of GMS is described in [15].

We suggest the construction of a FNN working as a classifier which predicts the event of a GMS four hours ahead. The structure of the network is constructed according to fuzzy IF—THEN rules obtained by an analysis of training samples.

The constructed system has been used for GMS prediction with the input parameters $n \cdot v$, σ_{B_z} (n is the plasma density of solar wind, v is the bulk velocity of solar wind, B_z is z-component of the interplanetary magnetic field, σ_{B_z} — the fluctuation of B_z).

It is known that the main phase of a GMS is usually preceded by significant changes of the parameters $n \cdot v$ and σ_{B_z}. The occurrence of the main phase can be explained by the following rule:

- *If a significant change of parameters $n \cdot v$ and σ_{B_z} occurs, then D_{st} index falls at least 40 nT in the next q hours.*

As a first step, we try to predict the GMS in the time interval $\langle t+1, t+q \rangle$ for $q = 4$ on the basis of the values of $n \cdot v(t)$ and $\sigma_{B_z}(t)$ or $B_z(t)$. We do not predict and approximate D_{st} value at the time t, but we analyze whether at the time interval $\langle t, t+q \rangle$, D_{st} falls at least 40 nT or not.

In this way it is possible to use NFC to predict GMS. We classify the addition $\Delta D_{\mathrm{st}}(t)$, $\Delta D_{\mathrm{st}}(t) = D_{\mathrm{st}}(t) - D_{\mathrm{st}}(t-2)$ to two classes C_B and C_N, where C_B is the class of falls with $\Delta D_{\mathrm{st}}(t) \leq -40$ nT and C_N is the supplement class of falls with $\Delta D_{\mathrm{st}}(t) > -40$ nT.

To prepare the training and testing samples we used the data from the years 1980–1984 and 1989–1992 because values of the parameters n, v, B_z, σ_{B_z} and D_{st} were measured at each hour and relatively few measured values are missing.

The data were prepared in the following way: if some value of the above parameters was missing in a particular hour then the data for this hour was omitted. The prepared data were represented by a sequence of $\boldsymbol{p}^t = (n^t, v^t, B_z^t, \sigma_{B_z}^t, D_{\mathrm{st}}^t)$, where $\boldsymbol{p}^t$ and $\boldsymbol{p}^{t+1}$ could not be measured in a 1 hour time difference.

In the next step, examples created from the sequence $\{\boldsymbol{p}^t\}_{t=1}^N$ were shown in the following form:

$$\left(n \cdot v^t, \sigma_{B_z}^t, \varphi(n \cdot v^t, \sigma_{B_z}^t)\right), \quad t = 2, \ldots, N-q,$$

where φ is *a classifying function* defined by (2)

$$\begin{gathered}
\varphi\left(n \cdot v^t, \sigma_{B_z}^t\right) = \begin{cases} 1 & \text{if } m_{t,q} \leq -40, \\ 0 & \text{otherwise}, \end{cases} \\
m_{t,q} = \min_{t'=t+1,\ldots,t+q} \left\{D_{\mathrm{st}}^{t'} - D_{\mathrm{st}}^{t'-2}\right\}.
\end{gathered} \tag{2}$$

The examples from the years 1980, 1981 and 1991 were used as the training samples, while the remaining examples were used for the testing.

Equation (2) implies that if the values of all the parameters are disposed at a time interval of q hours before the main phase of the GMS then all the q examples corresponding to the data have the third item 1. This means that samples consist of a group of examples with the third item 0 and of a (smaller) set of examples with the third item 1. One group with the third item 1 usually corresponds to one GMS. (However sometimes during one GMS there can be more falls in the course of few hours. If the time difference between the falls is greater than q hours then there are more groups of examples with the third item 1.)

Results of the application:. Criteria of a (non)success are two values E_1 and E_2 defined by equation (3),

$$E_1 = 1 - \frac{S}{B}\,, \qquad E_2 = \frac{M}{n}\,, \tag{3}$$

where

B – the actual number of GMSs in the sample. The value is equal to of groups with third item 1 (in the case without a reduction of examples with some missing data).

S – of GMSs correctly predicted by the network. The storm has been correctly predicted when the output of the network is 1 for at least one example from the group of examples with third item 1.

n – of examples in a testing sample with the third item 0.

M – of the examples with third item 0 for which the network gave the answer 1. This is of storms falsely predicted by the network.

E_1 expresses the nonsuccess of the network in the prediction of real GMSs, $1 - E_1$ is the success of the network in the prediction of real GMSs and E_2 expresses those examples with a fictive prediction of the storm. The learning algorithm applied to the initialized network minimizes the error E_2.

The three testing samples have been selected from the data measured during the years 1982–1984, 1989, 1990 and 1992 for both pairs of parameters.

Sample A contains 9975 examples from years 1982–1984. **Sample B** contains 7674 examples from years 1989, 1990 and 1992. **Sample C** contains 17649 examples from both above samples. Sample A contains 32, Sample B 94 and Sample C 126 GMSs.

Table 1. The results computed by neuro-fuzzy classifier with the parameters $n \cdot v$, σ_{B_z} after the learning process. The values E_2 are better than values in Table 2, but the values $1 - E_1$ are worse

			Sample A ($B = 32$)			Sample B ($B = 94$)			Sample C ($B = 126$)		
α	β	R	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2
0.3	20	74	17	53.12	0.91	33	35.11	1.56	50	39.68	1.18
0.35	20	75	16	50.00	0.97	29	30.85	1.86	45	35.71	1.34
0.25	20	100	16	50.00	1.07	26	27.66	2.20	42	33.33	1.55

The best three results are shown in Tables 1–2. Symbol R expresses of rules used in the classification and S is of correctly predicted GMSs. Instead of the values for the error E_1 in tables are the values of $1 - E_1 = \frac{S}{B}$ which express the success of NFC prediction. Both values are expressed as percentages. Tables 2 shows the results of the network initialized by the algorithm without the learning process. The first experiment shows that the applied learning process in both cases is improving E_2 (results of false prediction), however $1-E_1$ (the prediction accuracy) is going down.

TABLE 2. The results computed by a neuro-fuzzy classifier with the parameters $n \cdot v$, σ_{B_z} without the learning process. The values E_2 are worse than values in Table 1, but the values $1 - E_1$ are better

			Sample A ($B = 32$)			Sample B ($B = 94$)			Sample C ($B = 126$)		
α	β	R	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2	S	$1 - E_1$	E_2
0.3	20	74	19	59.38	1.57	36	38.30	2.44	55	43.65	1.93
0.35	20	75	19	59.38	1.78	39	41.49	2.95	58	46.03	2.27
0.25	20	100	17	53.12	2.19	39	41.49	4.40	56	44.44	3.11

6. Discussion and conclusion

We have presented the rule learning algorithm for a neuro-fuzzy classifier, which allows us to consider a rule basis that explains the properties of the problem better than the known constructions of the rule basis. The algorithm tries to effect the automatic rule learning process. The application that we discussed above shows that the algorithm can be applied in the prediction of GMS and similar problems.

Acknowledgement. The authors thank the referee for his remarks in evaluating this paper.

REFERENCES

[1] ANDREJKOVÁ, G.—VOJTÁŠ, P.: *Fuzzy prolog with neural networks connectives*, in: Proc. EUFIT '97, Verlag Maine Wissenschaftsverlag, Aachen, 1997, pp. 57–61.

[2] ANDREJKOVÁ, G.—TÓTH, H.—KUDELA, K.: *Fuzzy Neural Networks in the Prediction of Geomagnetic Storms*, in: Proceedings "Artificial Intelligence in STP", Lund, 1997, pp. 173–179.

[3] FULLÉR, R.: *Neural Fuzzy Systems*, Åbo Akademis tryckeri, 1995.

[4] GOTTWALD, S.: *Fuzzy Sets and Fuzzy Logic*, Vieweg-Verlag, 1993.

[5] ASSOUN, M. H.: *Fundamentals of artificial neural networks*, MIT Press, Cambridge, 1995.

[6] KŘUPKA, J.—OLEJ, V.: *Hierarchical Structure of Decision Processes on the Basis of DSP Starter Kit*, MENDEL '97, 3rd International Conference on Genetic Algorithms, Optimization problems, Fuzzy Logic, Neural Networks, Rough Sets, Brno, 1997, pp. 210–214.

[7] KVASNIČKA, V. et al.: *Introduction to the neural network theory*, IRIS, Bratislava, 1997.

[8] LIN, Y.—CUNNINGHAM, G.: *Building a Fuzzy System From Input-Output Data*, J. Intelligent & Fuzzy Systems, Vol. 2, No. 3, 1994, pp. 243–250.

[9] NAITO, E.—OZAWA, J.—HYASHI, I.—WAKAMI, N.: *A proposal of fuzzy connective with learning function and quaery networks for fuzzy retrieval systems*, in: In Fuzziness Database Management Systems, (P. Bosc and J. Kaczprzyk, eds.), Physica Verlag, 1995, pp. 345–364.

[10] NAUCK, D.—KRUSE, R.: *NEFCLASS – a neuro-fuzzy approach for the classification of data*, Applied Computing 1995, Proc. of the ACM Symposium on Applied Computing, Nashville, Feb 26–28, ACM Press, New York, 1995, pp. 461–465.

[11] NAUCK, D.—KRUSE, R.: *What are Neuro-Fuzzy Classifiers?*, in: Proceedings of 7-th IFSA World Congress, Prague, June 25–29, 1997, pp. 228–233.

[12] NAUCK, D.—KRUSE, R.: *New Learning Strategies for NEFCLASS*, in: Proceedings of 7-th IFSA World Congress, Prague, June 25–29, 1997, pp. 50–55.

[13] VAŠČÁK, J.—KOVÁČIK, P.—BETKA, F.: *Handling Results of a Fuzzy State Prediction Model Calculation by Sugeno Filter*, accepted for publication Proceedings FSTA '98, 4th International Conference on Fuzzy Sets Theory and its Applications, 1998.

[14] VOJTÁŠ, P.: *Fuzzy reasoning with tunable t-operators*, accepted for publication Journal of Advanced Computational Intelligence, Fuji Technology Press, Japan, 1998.

[15] WU, J. G.—LUNDSTEDT, H.: *Neural Networks Modelling of Solar Wind-Magnetosphere Interaction*, J. Geophys. Res. **102**, No. A7, 1997, pp. 14457–14466.

Received March 25, 1998

Department of Computer Science
Faculty of Science
P. J. Šafárik University
Jesenná 5
SK–041 54 Košice
SLOVAKIA

E-mail: andrejk@turing.upjs.sk
toth@duro.upjs.sk