TASK QUARTERLY 7 No 1 (2003), 7–22

FUZZY INFERENCE NEURAL NETWORKS WITH FUZZY PARAMETERS

DANUTA RUTKOWSKA[1] AND YOICHI HAYASHI[2]

[1] *Department of Computer Engineering, Technical University of Czestochowa, Armii Krajowej 36, 42-200 Czestochowa, Poland*
drutko@kik.pcz.czest.pl

[2] *Department of Computer Science, Meiji University, 1-1-1 Higashimita, Tama-ku, Kawasaki, 214-8571, Japan*
hayashiy@cs.meiji.ac.jp

(Received 5 May 2002)

Abstract: This paper concerns fuzzy neural networks and fuzzy inference neural networks, which are two different approaches to neuro-fuzzy combinations. The former is a direct fuzzification of artificial neural networks by introducing fuzzy signals and fuzzy weights. The latter is a representation of fuzzy systems in the form of multi-layer connectionist networks, similar to neural networks. Parameters of membership functions (centers and widths) play the role of neural network weights. In this paper, fuzzy inference neural networks with fuzzy parameters are considered. Neuro-fuzzy systems of this kind utilize both approaches: fuzzy neural networks and fuzzy inference neural networks. They also pertain to fuzzy systems of type 2 since membership functions with fuzzy parameters characterize type 2 fuzzy sets. Various architectures of these networks have been obtained for fuzzy systems based on different fuzzy implications. By analogy with fuzzy inference neural networks with crisp parameters, methods of learning fuzzy parameters and rule generation can be derived for neuro-fuzzy systems with fuzzy parameters. Fuzzy inference neural networks are studied in the framework of fuzzy granulation. In particular, fuzzy clustering as fuzzy information granulation is proposed to be applied in order to generate fuzzy IF-THEN rules. Applications of fuzzy inference neural networks are also outlined.

Keywords: neuro-fuzzy systems, fuzzy neural networks, fuzzy inference neural networks, fuzzy systems of type 2, fuzzy granulation

1. Introduction

Different approaches to neuro-fuzzy combinations have been considered in the literature [1, 2]. Direct fuzzification of neural networks by introducing fuzzy signals and fuzzy weights have been proposed in [3–6]. Neuro-fuzzy systems of this kind are called *fuzzy neural networks* (see also [7]). Their architectures are exactly the same as the connectionist multi-layer architectures of artificial neural networks [8], but they are fuzzy, as their connection weights as well as input and output values are fuzzy numbers. Other neuro-fuzzy systems, known as *fuzzy inference neural networks*,

are widely studied in the literature [9–12]. These are connectionist networks that represent fuzzy systems [13–15] in the form of multi-layer architectures, analogous to artificial neural networks. Instead of classical artificial neurons, the processing elements (nodes) of fuzzy inference neural networks perform various functions, for example, membership functions and the functions of T-norm or S-norm (T-conorm) operators. The architectures correspond to the type of fuzzy inference realized by the fuzzy systems represented by the networks.

The above mentioned connectionist fuzzy inference neural networks are not, in fact, fuzzy. The parameters of these networks that play the role of neural network weights, as well as input and output values, are crisp (not fuzzy) numbers. In this paper, fuzzy inference neural networks with fuzzy parameters are proposed, by analogy with fuzzy neural networks, which are neural networks with fuzzy weights. This approach can be viewed as a combination of both fuzzy inference neural networks and fuzzy neural networks. The neuro-fuzzy systems of this kind can be considered as the connectionist representation of fuzzy systems of type 2, since the fuzzy parameters refer to fuzzy sets of type 2, as defined in [16]. The crisp parameters of classical fuzzy inference neural networks are typical parameters (centers and widths) of fuzzy (type 1) sets introduced in [17].

2. Fuzzy neural networks

Fuzzy neural networks, with fuzzy signals and fuzzy weights, have the same connectionist forms as the corresponding classical multi-layer neural networks, where signals and weights are fuzzy numbers, usually triangular fuzzy sets. Neurons, which are processing elements in these networks, realize the same operations in both classical neural networks and fuzzy neural networks. They multiply signals by corresponding weights and add up the results. Transfer functions then change the results of this linear operation to neuron outputs. The transfer functions are most often sigmoidal functions.

Figure 1 illustrates a fuzzy neural network with one hidden layer. Of course, the network can contain more hidden layers, in addition to the input and output layers.

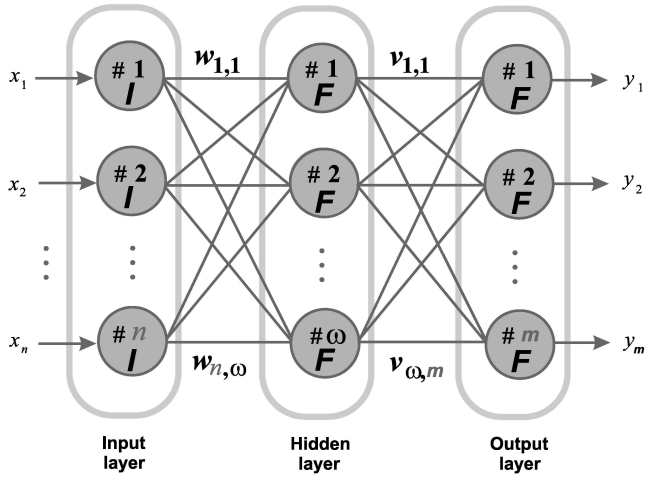

Figure 1. Fuzzy neural network

The input layer in Figure 1 includes n input neurons which only transmit input signals to their outputs, so they perform the identity function marked as I. Other neurons realize the transfer function denoted as F. In Figure 1, there are ω neurons in the hidden layer and m neurons in the output layer. The fuzzy weights associated with the connections between the input and the hidden layers are $w_{i,j}$, where $i=1,\ldots,n$ and $j=1,\ldots,\omega$. The fuzzy weights between the hidden and output layers are denoted as $\nu_{j,l}$, where $l=1,\ldots,m$.

There are basically two ways of computing output signals in fuzzy neural networks. One of them uses the extension principle introduced by Zadeh [17, 16], while another employs α-cuts and interval arithmetic (see [1] for details). Thus, fuzzy neural networks can be trained by means of a fuzzified version of the back-propagation algorithm, which is widely used for classical neural networks [8]. In [3–6], direct fuzzification of the back-propagation algorithm, called the fuzzified delta rule, has been applied. Methods of learning fuzzy neural networks are also described in [1].

3. Fuzzy inference neural networks

As mentioned in Section 1, fuzzy inference neural networks realize the inference process performed by a fuzzy system represented by a neuro-fuzzy connectionist architecture. A general form of multi-layer architecture is shown in Figure 2. This architecture reflects the mathematical formula that describes a fuzzy logic system with a singleton fuzzifier and a discrete form of the COA defuzzifier (*center of area* defuzzification method). This formula, which expresses the crisp output, $\bar{y}$, in the function of the crisp input, $\bar{x}$, is derived in [11, 12].

The first layer of the architecture portrayed in Figure 2 includes elements which refer to the antecedent fuzzy sets, *i.e.* the fuzzy sets in the antecedent part of the fuzzy IF-THEN rules used by the fuzzy logic system. These rules have the following form:

$$R^k : \textbf{IF } \mathbf{x} \text{ is } A^k \textbf{ THEN } y \text{ is } B^k, \tag{1}$$

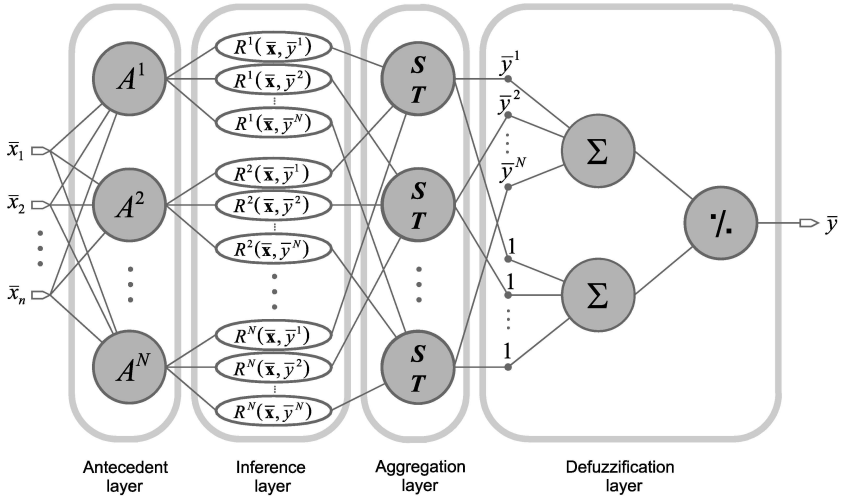

Figure 2. General form of neuro-fuzzy architecture

where $\mathbf{x} = [x_1, \ldots, x_n]^T \in \mathbf{X} \subset \mathbf{R}^n$ and $y \in \mathbf{Y} \subset \mathbf{R}$ are linguistic variables corresponding to the input and output of the system, $A^k = A_1^k \times \cdots \times A_n^k$ and B^k are fuzzy sets in the antecedent and consequent parts of the rules, respectively, for $k = 1, \ldots, N$, and N denotes the number of rules in the rule base.

If $x_1, \ldots, x_n$ are independent variables, then the rule base (1) can be written as follows:

$$R^k : \textbf{IF } x_1 \text{ is } A_1^k \text{ AND} \ldots \text{AND } x_n \text{ is } A_n^k \textbf{ THEN } y \text{ is } B^k. \tag{2}$$

Rule bases (1) and (2) refer to the MISO (multi-input, single-output) system. It is sufficient to consider this case, because results obtained for this kind of fuzzy (neuro-fuzzy) system can easily be extended to the MIMO (multi-input, multi-output) system.

Fuzzy IF-THEN rules (1) are interpreted as fuzzy relations $A^k \rightarrow B^k$, which are often called fuzzy implications. However a rationale for the latter name concerns only a fuzzy system based on a genuine implication in the logical sense, not systems based on the Mamdani approach, which are most often applied and considered in the literature [9, 14, 15]. In both kinds of systems, those based on the Mamdani and logical approaches, the inference process is performed according to the compositional rule of inference [16]. This is done using a fuzzy relation (implication) and the input fuzzy set which is generally a singleton. The second layer in Figure 2 refers to the inference process carried out by individual IF-THEN rules. Applying a singleton fuzzifier means that the input fuzzy sets are characterized by membership functions which take the value equal to 1 for the crisp input, $\bar{\mathbf{x}} = [\bar{x}_1, \ldots, \bar{x}_n]^T \in \mathbf{R}^n$, and the value equal to 0 when the input values differ from $\bar{\mathbf{x}}$. In this case, the membership function of a fuzzy set inferred by an individual IF-THEN rule equals to the membership function of the fuzzy set $A^k \rightarrow B^k$ for $\mathbf{x} = \bar{\mathbf{x}}$. Thus, the elements of the inference layer in Figure 2 perform these membership functions for $k = 1, \ldots, N$.

The third layer of the network shown in Figure 2 is the aggregation layer. It contains the elements which realize the S-norm or T-norm operation, depending on whether the Mamdani or logical approach is employed. In the former approach, the S-norm operator is used in order to aggregate the fuzzy sets inferred by the individual IF-THEN rules. In the latter approach, the T-norm operator is applied, resulting in an aggregated output fuzzy set. Usually, the max and min operators are chosen as the S-norm and T-norm, respectively. The proper name for an S-norm is a T-conorm, however the name of S-norm is often used [18].

The last part of the multi-layer architecture, called the defuzzification layer, reflects the defuzzifier which maps fuzzy sets in $\mathbf{Y} \subset \mathbf{R}$ to crisp points in $\mathbf{Y}$. As mentioned earlier, the discrete form of the COA defuzzification method has been employed. This kind of defuzzifier is described by the following equation:

$$\bar{y} = \frac{\sum_{k=1}^{N} \bar{y}^k B'\left(\bar{y}^k\right)}{\sum_{k=1}^{N} B'(\bar{y}^k)}, \tag{3}$$

where $\bar{y}$ is the crisp output value of the system, $\bar{y}^k$ is the center of the membership function of fuzzy set B^k, that is the point with the maximal value of this membership

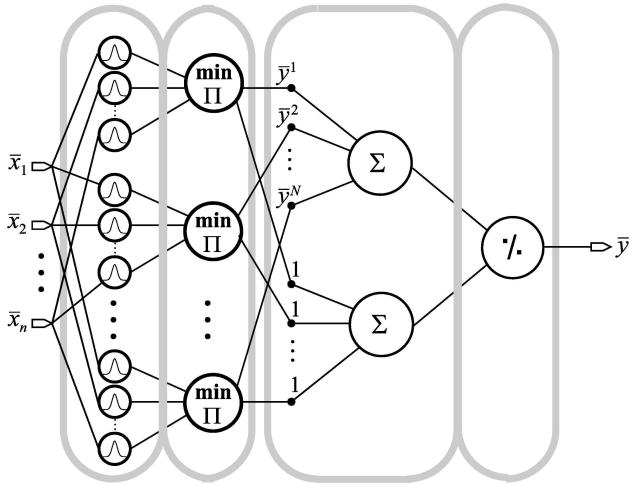

Figure 3. Neuro-fuzzy architecture of a system based on the Mamdani approach

function, B' is the aggregated fuzzy set, and – in this formula – the same notation is used for the membership function of B'.

Figure 3 illustrates a special case of the general neuro-fuzzy architecture presented in Figure 2, for a system based on the well-known Mamdani approach to fuzzy inference. The first two layers in the network depicted in Figure 3 correspond to the antecedent layer in Figure 2. The last two layers in Figure 3 constitute the defuzzification layer portrayed in Figure 2. Assuming that each layer includes the same kind of elements (nodes, neurons), the defuzzification layer in Figure 2 actually consists of the two layers shown in Figure 3. There are two classical linear neurons in the first defuzzification layer and only one element performing the division operation in the last layer. The defuzzification layers reflect formula (3).

The first part of the fuzzy inference network illustrated in Figure 3 refers to fuzzy IF-THEN rules (2). The elements of the first layer realize the membership functions of the antecedent fuzzy sets $A_1^k,\ldots,A_n^k$, for $k=1,\ldots,N$. The Gaussian membership functions are marked in Figure 3, however other types, *e.g.* triangular functions, can be applied. The next layer contains the elements which perform the min or product operation, depending on the operation chosen to realize the Cartesian product $A_1^k \times \cdots \times A_n^k$.

The architecture of the network portrayed in Figure 3 is determined in [9], assuming a singleton fuzzifier and CA (*center average*) defuzzification method. This neuro-fuzzy architecture represents a fuzzy logic system based on the Mamdani approach. The system of this kind employs the min or product operation as the fuzzy relation $A^k \rightarrow B^k$ and refers to the Mamdani or Larsen rule of inference, respectively. Using these simple operations and the assumption that the consequent fuzzy sets, B^k, for $k=1,\ldots,N$, are non-overlapping, it is easy to obtain the network depicted in Figure 3 from the general architecture shown in Figure 2. For details, see [11, 12].

4. Implication-based neuro-fuzzy architectures

In Section 3, a special case of fuzzy inference neural networks, derived from the general neuro-fuzzy architecture illustrated in Figure 2 has been described. This network represents a fuzzy logic system based on the Mamdani approach. Employing various logical implications to the neuro-fuzzy systems in the form of architecture shown in Figure 2, different multi-layer networks for systems based on the logical approach can be determined. Architectures of this kind are described in [11, 12]. These architectures differ with respect to the inference layer. The architectures are simpler for non-overlapping consequent fuzzy sets (NOCFS) and more complicated when the overlapping consequent fuzzy sets (OCFS) are considered. The inference layers contain the elements that realize the min and product operations, as well as others, *e.g.* negation, max, summation. Simpler networks have only one inference layer, the more complex architectures include several layers of different elements.

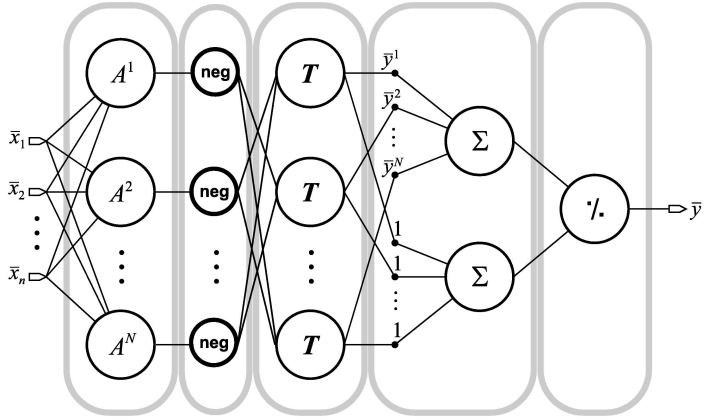

Figure 4. An example of neuro-fuzzy architecture based on logical approach

Figure 4 portrays the neuro-fuzzy architecture of a system based on the Kleene-Dienes group of implications, in the case of NOCFS. According to [11], the following implications are examples of those belonging to the Kleene-Dienes group of implications: Kleene-Dienes, Łukasiewicz, Reichenbach, stochastic, Dubois-Prade, and Fodor. Fuzzy inference systems based on these implications have the same multi-layer architecture, shown in Figure 4, in the NOCFS case. However, the OCFS neuro-fuzzy architectures are different for each implication from this group. For details, see [11, 12].

The architecture depicted in Figure 4 represents a system described by the following equation:

$$\bar{y} = \frac{\sum_{k=1}^{N} \bar{y}^k \underset{\substack{j=1 \\ j\neq k}}{\overset{N}{\boldsymbol{T}}} \left(1 - A^j(\bar{\mathbf{x}})\right)}{\sum_{k=1}^{N} \underset{\substack{j=1 \\ j\neq k}}{\overset{N}{\boldsymbol{T}}} \left(1 - A^j(\bar{\mathbf{x}})\right)}, \tag{4}$$

where $\boldsymbol{T}$ denotes the T-norm operator of N arguments, usually chosen as the min or product operator, and $A^j(\bar{\mathbf{x}})$ is the membership function of fuzzy set A^j, for $j = 1, \ldots, N$, at the point $\bar{\mathbf{x}}$, so $(1 - A^j(\bar{\mathbf{x}}))$ is the value of the membership function of the **negation** of fuzzy set A^j at the point $\bar{\mathbf{x}}$. The negation operation is performed by

the elements of the second layer of the architecture illustrated in Figure 4. Formula (4) is derived in [11, 12].

Both kinds of fuzzy inference neural networks, those based on the Mamdani and logical approaches, can be trained in a way similar to that of applied to classical multi-layer neural networks, *i.e.* using the idea incorporated into the back-propagation learning algorithm (see *e.g.* [8]). This idea comes from the steepest descent optimization method [19] and can also be used in order to find or tune parameters of the neuro-fuzzy systems. In this way, the recursive procedures, analogous to those applied for adjusting weights in neural networks, constitute learning algorithms for the particular neuro-fuzzy architectures. However, it is not necessary to determine these mathematical expressions for each neuro-fuzzy system. It is possible to perform this kind of learning based on the architecture, without knowing the recursive formulas [20]. The FLiNN software [21, 22] is an example of a computer program which realizes the learning of this type. This program creates the architecture composed of the elements, properly connected. Then, it conducts the error back-propagation through the elements of the network in such a way that each element propagates the error signal from its output to the outputs of the elements in the preceding layer. This method allows to find weights of a classical artificial neural network or parameters (centers and widths of membership functions) of a fuzzy inference neural network.

5. Fuzzy sets and fuzzy systems of type 2

The fuzzy sets defined by Zadeh in [17] are characterized by membership functions which associate with each point (member of the fuzzy set) its grade of membership, expressed by a real number in the interval $[0,1]$. In this case, the membership grades are precise (crisp) numbers. These fuzzy sets can be called fuzzy sets of type 1. The concept of a type 2 fuzzy set, as well as higher type fuzzy sets, was introduced by Zadeh [16] to deal with situations where uncertainty can exist about the membership grades themselves. A type 1 fuzzy set is a special case of a type 2 fuzzy set. The definition formulated in [23] states the following: *A fuzzy set of type 2 is defined by a fuzzy membership function, the grade (that is, fuzzy grade) of which is a fuzzy set in the unit interval* $[0,1]$*, rather than a point in* $[0,1]$.

An interested reader can find more detailed, formal explanations concerning fuzzy sets of type 2 in the literature [23–25].

According to the definition proposed in [16], the membership function of a fuzzy set of type 1 ranges over the interval $[0,1]$, the membership function of a fuzzy set of type 2 ranges over fuzzy sets of type 1, the membership function of a fuzzy set of type 3 ranges over fuzzy sets of type 2, *etc.*, for fuzzy sets of type 4, 5, ...

Interval type 2 sets, which are the simplest kind of type 2 fuzzy sets, have also been considered in the literature [26–28].

To define operations on fuzzy sets of type 2, it is natural to make use of the extension principle and interval-valued membership functions (see [16]). More details with regard to operations on type 2 fuzzy sets can be found in [29].

Fuzzy sets of type 2 have been applied to fuzzy systems (see *e.g.* [30–32]). In order to simplify the computations concerning the inference performed by the system, we can assume that secondary fuzzy sets are interval sets [32]. In this case, values of the

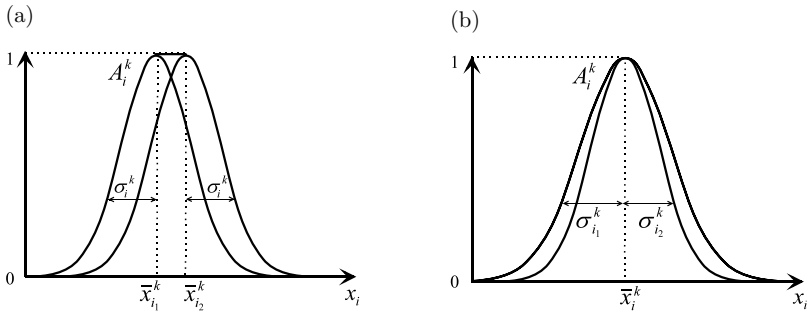

Figure 5. Gaussian membership functions with interval type fuzzy parameters

secondary membership functions are either one or zero. Figure 5 illustrates a Gaussian membership function of this type.

Figures 5a and 5b show a Gaussian membership function with fuzzy (interval) center and width, respectively. The center can vary from $\bar{x}_{i_1}^k$ to $\bar{x}_{i_2}^k$, and width from $\sigma_{i_1}^k$ to $\sigma_{i_2}^k$. The narrow area between two Gaussian functions, in both figures, represents the uncertainty of the parameters.

6. Fuzzy inference neural networks with type 2 fuzzy sets

Fuzzy neural networks, considered in Section 2, are neural networks with fuzzy weights and signals. They can be trained using a fuzzified version of the classical back-propagation learning algorithm. The fuzzy inference neural networks described in Sections 3 and 4 are not, in fact, fuzzy networks. Their architectures contain elements that process crisp (real-valued) signals and are characterized by crisp parameters. The main difference between weights of classical neural networks and parameters of fuzzy inference neural networks is that the latter can be interpreted as centers and widths of membership functions. Thus, the knowledge is in the form of IF-THEN rules, while the knowledge of classical neural networks is stored in their weights and does not explain the network performance unless the rules are extracted from the network.

In this paper, fuzzy inference neural networks in the form of various multi-layer architectures are treated analogically to fuzzy neural networks, by introducing fuzziness to their parameters. Thus, the elements of the first layer of these architectures realize membership functions with fuzzy parameters (centers and widths), which means that fuzzy sets of type 2 are used (see Section 5). An example of membership functions of this type is shown in Figure 5. The consequent fuzzy sets can be of type 1 or type 2. When the fuzzy system is applied to a control problem, the latter case can be considered, however the former is suitable for most of the classification tasks.

Since fuzzy inference neural networks represent fuzzy logic systems with a fuzzifier (singleton) and a defuzzifier (COA or CA), we can assume that the inputs are crisp values. However, if the parameters are fuzzy, the output values are also fuzzy. Therefore, another defuzzification is required for the fuzzy output of the network. It is worth mentioning that in type 2 fuzzy logic systems [31], apart from a defuzzifier, the so-called *type-reducer* is employed in order to reduce the output fuzzy set of type 2 to a fuzzy set of type 1. The simplest method that we can propose is to take centers of the fuzzy outputs as the corresponding crisp outputs. This means that the crisp

values with the highest membership grades are considered as the outputs of the system. Other methods of type-reduction can be found in [31]. Figure 6 portrays a fuzzy inference neural network (with type 2 fuzzy sets) that corresponds to a fuzzy logic system with a fuzzifier and a defuzzifier. A type-reducer is added to the network.

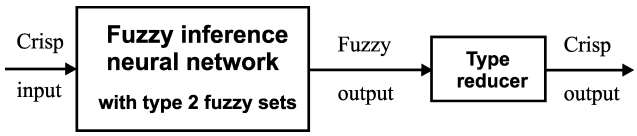

Figure 6. Fuzzy inference neural network with a type-reducer

In type 2 fuzzy logic systems, a type-reducer can be included before a defuzzifier [31] or the type-reduction is an extended version of a type 1 defuzzification method, by use of the extension principle [16, 17]. This operation is called type-reduction, as it reduces the type of the output fuzzy set (from type 2 to type 1). It is worth emphasizing that in many applications the type-reduced set is more significant than a single crisp output value since it conveys a measure of uncertainties in the type 2 fuzzy system [32].

7. Learning methods of fuzzy parameters

As mentioned in Sections 2 and 6, fuzzy neural networks can be trained using fuzzified versions of the algorithms employed as learning methods for classical neural networks, *e.g.* the back-propagation algorithm. Fuzzy arithmetic can be applied to fuzzify the classical learning methods.

In order to train fuzzy inference neural networks, recursive learning formulas can be derived from the steepest descent optimization method, resulting in algorithms similar to those of the back-propagation method of learning neural networks [33]. Since the classical back-propagation algorithm can be fuzzified, it seems obvious that fuzzified versions of the recursive procedures for learning fuzzy inference neural networks can be determined in the similar way. However, as emphasized in Section 4, it is not necessary to know these procedures. It is possible to train neural networks as well as fuzzy inference neural networks based on their multi-layer architectures. Thus, instead of formulating mathematical recursions corresponding to the fuzzified learning algorithms, we can use fuzzy arithmetic in order to propagate signals (errors) through the elements of the architectures with fuzzy parameters, according to the idea of the steepest descent and back-propagation methods. First of all, a library of the basic elements that are components of various networks should be created. It is important that every element "knows" how to propagate the fuzzy signals from its inputs to the output and from its output to the outputs of the elements in the preceding layer. With this library of the elements, it may be possible to employ this kind of learning to various architectures with fuzzy parameters. Of course, a fuzzified version of the software that works similarly to the FLiNN program [21] must be developed.

Competitive learning, proposed in [34], can also be used in order to train fuzzy inference neural networks. The FLiNN software may be employed to perform this kind of learning. Competitive learning is applicable to the first layer, which is the same in every architecture. Figure 7 shows the elements of this layer, assuming that

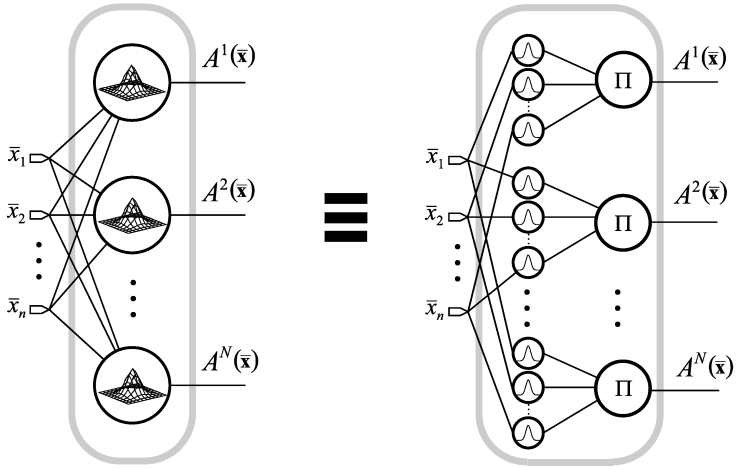

Figure 7. First layer of inference neural networks

Gaussian membership functions are applied and the Cartesian product is defined by the product operation.

The elements of the second part of Figure 7 which realize Gaussian membership functions can incorporate fuzzy parameters as presented in Figure 5. Using the idea of competitive learning, values of the membership functions $A^j(\bar{\mathbf{x}})$, for $j = 1,\ldots,N$, at the output of the first layer shown in Figure 7, are taken into account. Then, only those parameters which correspond to the maximal output value $A^j(\bar{\mathbf{x}})$ are modified according to the learning algorithm. It is worth emphasizing that the methods which reduce learning complexity are very important, especially in the case of type 2 fuzzy sets.

8. Rule generation

The idea of learning discussed in Section 7 can be used in order to adjust fuzzy parameters of fuzzy inference neural networks, assuming that the rule base in the form (1) or (2), with the fuzzy sets of type 2, is known. Thus, the learning method suggested, based on the network architectures, can tune the fuzzy parameters, but the architectures are constructed using the knowledge base composed of the fuzzy IF-THEN rules.

Like in the cases of fuzzy inference neural networks with crisp parameters, the rule base can be given for a problem under consideration, but there are many tasks for which we have learning data and do not know the rules. Thus, various methods are developed in order to generate fuzzy IF-THEN rules from numerical data (see *e.g.* [35, 36], as well as the survey paper [37]).

We can try to adopt the existing algorithms of rule generation for fuzzy inference neural networks with fuzzy parameters. Type 2 fuzzy sets have already been introduced to fuzzy and neuro-fuzzy systems, and to learning methods, for example, carried out by clustering [38].

Clustering methods can be applied to rule generation, but the number of clusters, which corresponds to the number of rules, usually needs to be fixed (see the well-known fuzzy *c-means* algorithm in [39]). Therefore, some other techniques

are employed to find the number of rules. Two recently developed algorithms for rule generation, applicable to fuzzy inference neural networks with crisp parameters, are described in [11]. These methods combine some ideas incorporated in the classical clustering algorithms with some heuristic techniques in order to generate the correct number of IF-THEN rules (see also [40, 41]). As results, we obtain crisp values of the membership function parameters, which define the fuzzy sets in the rule base. If necessary, the values of these parameters can be tuned by use of the gradient method implemented in the FLiNN program (see Section 4).

In order to generate fuzzy IF-THEN rules with fuzzy parameters, we propose to extend the methods mentioned above so that fuzzy prototypes (*e.g.* cluster centers which correspond to the centers of the membership functions) will be determined. In this case, fuzzy clusters are treated as fuzzy sets of type 2, so fuzzy arithmetic with operations on type 2 fuzzy sets can be used. Of course, other rule generation methods may be adopted in a similar way for fuzzy inference neural networks with fuzzy parameters.

The situation when fuzzy rules are generated from a sequence of input-output data and then parameters of membership functions are adjusted based on the training data can be viewed as hybrid learning composed of two stages: rule generation and parameter tuning (see Figure 8). It is worth mentioning that the FLiNN program can also generate fuzzy IF-THEN rules by means of the method proposed in [35], then create the architecture based on these rules and adjust parameters of the membership functions.

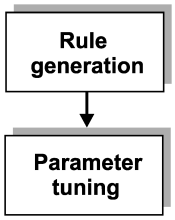

Figure 8. Hybrid learning of fuzzy inference neural networks

9. Fuzzy granulation

According to Zadeh [42, 43], linguistic variables, used in fuzzy IF-THEN rules (1) or (2), are concomitant with the concept of granulation. In fuzzy logic, granulation involves a grouping of objects into fuzzy granules, with a granule being a clump of objects drawn together by similarity. In effect, granulation may be viewed as a form of fuzzy quantization, which in turn may be seen as an instance of fuzzy data compression. Values of a linguistic variable may be treated as granules whose labels are the linguistic values of the variable. The foundation of the theory of fuzzy information granulation, the basis of *computing with words* [44, 45], comes from the concept of linguistic variables and fuzzy IF-THEN rules.

As a matter of fact, the theory of fuzzy information granulation with the *calculus of fuzzy graphs*, leads to the concept of computing with words, which involves manipulation of words rather than numbers. A word is assumed to be a label of a fuzzy granule, so words may be viewed as forms of fuzzy granulation.

The concept of a fuzzy graph was introduced in [13], and developed *e.g.* in [16]. Then, the so-called calculus of fuzzy graphs was considered (see *e.g.* [43]). The concept of fuzzy graphs plays a key role in fuzzy logic and is employed in most of its applications. The theory of fuzzy information granulation also underlines fuzzy graphs.

A collection of fuzzy IF-THEN rules can be represented by a fuzzy graph. Using the definition of the intersection of fuzzy graphs [46], the inference process of fuzzy systems, based on the compositional rule of inference, may be illustrated by means of fuzzy graphs. A fuzzy graph can portray a relation which corresponds to a collection of rules as a disjunctive representation of fuzzy points which are Cartesian products of fuzzy sets.

Figure 9 is an illustration of a fuzzy graph that corresponds to a collection of fuzzy IF-THEN rules, and vice versa. The fuzzy graph, f^*, approximates the dependency given by function f which is coarsely described by the rules. Fuzzy granulation of the domains of variables x and y, where fuzzy granules are the linguistic values employed in fuzzy IF-THEN rules, constitutes the basis on which the fuzzy graph is created.

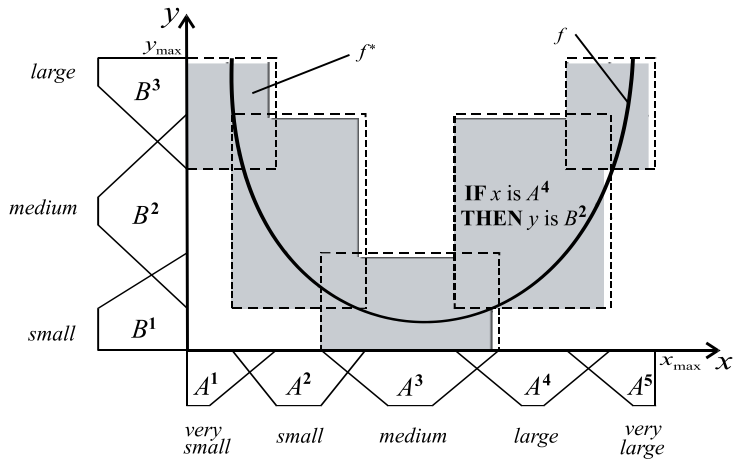

Figure 9. Illustration of a fuzzy graph

The fuzzy graph portrayed in Figure 9 refers to the meaning of an elementary IF-THEN rule, expressed as follows:

$$\textbf{IF } x \text{ is } A \textbf{ THEN } y \text{ is } B \rightarrow (x,y) \text{ is } A \times B, \tag{5}$$

where "→" should be read as "translates into", and $A \times B$ denotes the Cartesian product of fuzzy sets A and B (see *e.g.* [11]).

The collection of N fuzzy IF-THEN rules is presented as:

$$\textbf{IF } x \text{ is } A^k \textbf{ THEN } y \text{ is } B^k \rightarrow (x,y) \text{ is } \sum_{k=1}^{N} A^k \times B^k, \tag{6}$$

where the summation denotes disjunction.

The Cartesian product $A^k \times B^k$, for $k = 1,\ldots,N$, is treated as a fuzzy point of the fuzzy graph which is viewed as a disjunctive superposition of the fuzzy points.

The meaning of fuzzy rules expressed by formulas (5) and (6) refers to the Mamdani approach. Another interpretation of fuzzy IF-THEN rules, with regard to the logical approach, based on genuine logical implications, can be found in [46] and [11].

It is stated in [47] that type 2 fuzzy sets and type 2 fuzzy logic are more suitable for computing with words than type 1 fuzzy sets and fuzzy logic.

It is easy to notice that clusters produced by clustering methods can be viewed as granules and fuzzy clusters obtained by fuzzy clustering algorithms are fuzzy granules. By clustering, we usually mean the partitioning of a collection of objects (data) into subsets, called clusters, that contain elements with common properties which distinguish them from the members of the other clusters. Thus, the elements within each cluster should be as similar to each other as possible and dissimilar from those of other clusters.

The fuzzy inference neural networks with fuzzy parameters presented in this paper may be considered in the framework of fuzzy granulation. The concept of fuzzy graphs can be used in order to illustrate and analyze the inference process performed by the networks.

It is worth mentioning that neural networks in the framework of granular computing are studied in [48], where clustering is also regarded as a synonym of information granulation.

10. Applications

There are many and varied problems which can be solved by the use of classical neural networks, fuzzy neural networks, fuzzy logic systems and fuzzy inference neural networks. These problems are usually concerned with function approximation, classification, control and other applications, *e.g.* prediction. Expert systems are created in the form of classical and fuzzy neural networks, fuzzy systems, and neuro-fuzzy systems [1, 3, 8, 11, 49]. Fuzzy inference neural networks with fuzzy parameters can thus be applied to the same type of tasks.

In [50] the neuro-fuzzy systems called NEFCON, NEFCLASS, and NEFPROX are described. These systems have been designed for control, classification and approximation problems, respectively. Their connectionist architectures are in the form of a generic fuzzy multi-layer perceptron. These networks have fuzzy weights and reflect fuzzy IF-THEN rules. The learning algorithms of these systems consist of two phases: learning fuzzy rules (rule generation) and learning fuzzy sets (parameter learning). These neuro-fuzzy systems do not employ fuzzy sets of type 2. However, it is shown in [51] that these systems can be considered as fuzzy inference neural networks (also with fuzzy parameters). An equivalence between both kinds of the neuro-fuzzy systems is illustrated, and the concept of type 2 fuzzy neural networks is proposed. This means that neural or neuro-fuzzy networks of type 2 can be transformed into their equivalent networks of type 1 (and vice versa). This is very important with regard to learning methods.

In order to test the systems or networks for control applications, well-known examples such as the *inverted pendulum* and *the truck backer-upper control problem* are used (see *e.g.* [9, 11, 22]). Various classification tasks are considered in the

literature, including the most common IRIS classification (for details see *e.g.* [11, 39, 52]). Some approximation problems are also illustrated in [11].

As mentioned in Section 6, fuzzy inference neural networks with type 2 fuzzy sets can be applied to control as well as classification tasks. Of course, these neuro-fuzzy systems can also solve approximation and other problems. It seems that, in the case of classification, consequent fuzzy sets of type 1 can be used. However, in the framework of fuzzy granulation, we can also consider type 2 fuzzy sets in the consequent part of the rules.

Fuzzy inference neural networks can be applied to various medical diagnosis problems, in particular, as expert systems (see *e.g.* [11, 12, 33, 41]). We can expect some advantages of employing the neuro-fuzzy systems with fuzzy parameters in medical diagnosis, especially when fuzzy IF-THEN rules are generated from medical data.

In [53] a medical diagnosis problem is solved by means of a parallel processing neuro-fuzzy system that employs various fuzzy implications. The results show that the optimal parameters of fuzzy sets do not need to be crisp but can be fuzzy, which means that a neuro-fuzzy system of type 2 can be used.

Promising applications of type 2 fuzzy logic systems are observed in [32]. Type 2 fuzzy systems are much more robust to noise than type 1 systems, because they are suitable for handling uncertainties. They outperform type 1 fuzzy systems in chaotic time-series prediction. It is believed that type 2 systems may be advantageous over their type 1 counterparts in the areas of pattern recognition, mobile communications, data mining, and others.

11. Conclusions

In this paper, fuzzy inference neural networks that represent fuzzy systems of type 2 are viewed as fuzzy inference neural networks with fuzzy parameters, by analogy with fuzzy neural networks which are fuzzified versions of classical neural networks (with crisp weights). Therefore, it is suggested to extend methods of learning of fuzzy inference neural networks with crisp parameters to the algorithms suitable for networks with fuzzy parameters. This should be done similarly to the way in which fuzzified versions of learning methods for fuzzy neural networks have been derived from the algorithms known for classical neural networks.

On the other hand, it seems that the learning methods used in order to train networks of type 1 can be employed to the equivalent networks of type 2. As mentioned in Section 10, an equivalence between the NEFCON, NEFCLASS, and NEFPROX systems (which are fuzzy neural networks) and fuzzy inference neural networks (which are in fact not fuzzy networks) is shown in [51]. Thus, instead of using a learning method appropriate for a system of type 2, we can transform this network to its equivalent of type 1 and then apply a suitable algorithm to that network.

Apart from parameter learning, rule generation methods for fuzzy inference neural networks with fuzzy parameters can be proposed based on fuzzy clustering. The networks are considered in the framework of fuzzy granulation, and fuzzy graphs can be employed to illustrate fuzzy IF-THEN rules as well as the inference process.

References

[1] Buckley J J and Feuring T 1999 *Fuzzy and Neural: Interactions and Applications*, Springer-Verlag, Heidelberg, New York

[2] Takagi H 2000 *Int. J. Appl. Math. Comp. Sci.* **10** (4) 647

[3] Buckley J J and Hayashi Y 1992 *Fuzzy Sets and Artificial Intelligence* **1** 11

[4] Buckley J J and Hayashi Y 1994 *Fuzzy Sets and Systems* **66** 1

[5] Buckley J J and Hayashi Y 1994 *Fuzzy Sets, Neural Networks and Soft Computing* (Zadeh L A and Yager R R, Eds.), Van Nostrand Reinhold, New York, pp. 233–249

[6] Hayashi Y, Buckley J J and Czogała E 1993 *Int. J. Intelligent Systems* **8** 527

[7] Rutkowska D and Hayashi Y 1999 *J. Adv. Comput. Intelligence* **3** (3) 177

[8] Żurada J M 1992 *Introduction to Artificial Neural Systems*, West Publishing Company

[9] Wang L-X 1994 *Adaptive Fuzzy Systems and Control*, PTR Prentice Hall, Englewood Cliffs, New Jersey

[10] Rutkowska D 2000 *Fuzzy Control Theory and Practice* (Hampel R, Wagenknecht M and Chaker N, Eds.), Springer-Verlag, Heidelberg, New York, pp. 277–286

[11] Rutkowska D 2002 *Neuro-Fuzzy Architectures and Hybrid Learning*, Springer-Verlag, Heidelberg, New York

[12] Rutkowska D and Nowicki R 2000 *Int. J. Appl. Math. Comp. Sci.* **10** (4) 675

[13] Zadeh L A 1971 *Aspects of Network and System Theory* (Kalman R E and DeClaris N, Eds.), Holt, Rinehart and Winston, New York, pp. 209–245

[14] Dubois D and Prade H 1980 *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, London

[15] Pedrycz W 1993 *Fuzzy Control and Fuzzy Systems*, Wiley, New York

[16] Zadeh L A 1975 *Information Science* Part I **8** 199; Part II **8** 301; Part III **9** 43

[17] Zadeh L A 1965 *Information and Control* **8** (3) 338

[18] Klement E P, Mesiar R and Pap E 2000 *Triangular Norms*, Kluwer, Dordrecht, Boston, London

[19] Curry H 1944 *Quart. Appl. Math.* **2** 258

[20] Piliński M 1996 *Proc. 2nd Conf. Neural Networks and Their Applications*, Szczyrk, Poland, pp. 383–391

[21] Piliński M 1997 *FLiNN – User Mannual*, Polish Neural Network Society, Czestochowa, Poland (in Polish)

[22] Rutkowska D, Piliński M and Rutkowski L 1997 *Neural Networks, Genetic Algorithms, and Fuzzy Systems*, PWN, Warsaw, Poland (in Polish)

[23] Mizumoto M and Tanaka K 1976 *Information and Control* **31** 312

[24] Mizumoto M and Tanaka K 1981 *Fuzzy Sets and Systems* **5** 277

[25] John R 1998 *Proc. 7th IEEE Int. Conf. Fuzzy Systems (FUZZ-IEEE'98)*, Anchorage, AK, USA, pp. 1003–1008

[26] Hisdal E 1981 *Int. J. Man-Machine Studies* **15** 385

[27] Gorzałczany M 1989 *Fuzzy Sets and Systems* **31** 243

[28] Türksen I B 1995 *Advances in Fuzzy Theory and Technology* (Wang P, Ed.), Duke Univ., Durhan, NC, USA, pp. 31–81

[29] Karnik N N and Mendel J M 1998 *University of Southern California Report*, http://sipi.usc.edu/~mendel/report

[30] Türksen I B 1999 *Fuzzy Sets and Systems* **106** 11

[31] Karnik N N, Mendel J M and Liang Q 1999 *IEEE Trans. Fuzzy Syst.* **7** (6) 643

[32] Liang Q and Mendel J M 2000 *IEEE Trans. Fuzzy Syst.* **8** (5) 535

[33] Rutkowska D 1997 *Intelligent Computational Systems. Genetic Algorithms and Neural Networks in Fuzzy Systems*, PLJ Academic Publishing House, Warsaw, Poland (in Polish)

[34] Nowicki R and Rutkowska D 2001 *Proc. East West Fuzzy Colloquium 2001*, Zittau, Germany, pp. 207–213

[35] Wang L-X and Mendel J M 1992 *IEEE Trans. Syst., Man, Cybern.* **22** (6) 1414

[36] Ishibushi H, Nozaki K, Yamamoto N and Tanaka H 1995 *IEEE Trans. Fuzzy Syst.* **3** 260

[37] Mitra S and Hayashi Y 2000 *IEEE Trans. Neural Networks* **11** (3) 748
[38] John R, Innocent P R and Barnes M R 1997 *Proc. 3rd Joint Conf. Information Science* **1** 58
[39] Bezdek J C 1981 *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York
[40] Rutkowska D and Starczewski A 1999 *Proc. 4th Conf. Neural Networks and Their Applications*, Zakopane, Poland, pp. 220–225
[41] Rutkowska D and Starczewski A 2000 *Fuzzy Systems in Medicine* (Szczepaniak P S, Lisboa P J G and Kacprzyk J, Eds.), Springer-Verlag, Heidelberg, New York, pp. 503–518
[42] Zadeh L A 1979 *Advances in Fuzzy Set Theory and Applications* (Gupta M, Ragade R and Yager R, Eds.), North Holland, Amsterdam, pp. 3–18
[43] Zadeh L A 1997 *Fuzzy Sets and Systems* **90** 111
[44] Zadeh L A 1996 *IEEE Trans. Fuzzy Syst.* **4** 103
[45] Zadeh L A 1999 *IEEE Trans. Circ. Syst. – I: Fundamental Theory and Applications* **45** (1) 105
[46] Zadeh L A 1996 *Multiple Valued Logic* **1** 1
[47] Mendel J M 1999 *Proc. 3rd Int. ICSC Symposium on Fuzzy Logic and Applications*, Rochester Univ., Rochester, NY, USA, pp. 158–164
[48] Pedrycz W 2000 *Int. J. Appl. Math. Comp. Sci.* **10** (4) 723
[49] Hayashi Y 1992 *Proc. 1st IEEE Int. Conf. Fuzzy Systems (FUZZ-IEEE'92)*, San Diego, CA, USA, pp. 485–491
[50] Nauck D, Klawonn F and Kruse R 1997 *Foundations of Neuro-Fuzzy Systems*, Wiley, New York
[51] Rutkowska D 2002 *Proc. IEEE Int. Conf. Fuzzy Systems (FUZZ-IEEE'02)*, The 2002 IEEE World Congress on Computational Intelligence, Honolulu, Hawaii, pp. 1180–1185
[52] Czogała E and Łęski J 2000 *Fuzzy and Neuro-Fuzzy Intelligent Systems*, Springer-Verlag, Heidelberg, New York
[53] Rutkowska D, Nowicki R and Hayashi Y 2002 *Lecture Notes in Computer Science* **2328** 599