Abstract

A fuzzy rule based system equivalent to an ANN is presented in this work. The inputs of this system are the input variables of the ANN and it uses fuzzy unions in the if-part of their fuzzy rules. Thus, the knowledge inside this system is comprehensible.

Keywords: Neural Networks, Fuzzy systems.

1 Introduction

Artificial Neural Networks (ANNs) [7, 13, 19] are computational models that have been successfully used in many areas such as automatic control, weather forecasting, etc. However, they suffer from a shortcoming, it is difficult to understand how an ANN solves a problem. Many works have tried to overcome this shortcoming. A review can be found in [1, 5, 8, 14].

Some authors have directed their efforts to find a Fuzzy Rule Based System (FRBS) [10, 23] equivalent to an ANN [2, 3, 11]. In this manner, the fuzzy rules, stated in natural linguistic terms, can provide some comprehension about the behavior of ANN.

In [11], it is presented FRBSs with “and” logical operator in the if-part of each rule extracted from ANNs. However, the inputs of this fuzzy system are different from the ones of the ANN. The inputs \( z_j \) of the fuzzy system have the format \( z_j = \sum_i (x_i w_{ij}) \), where the variables \( x_i \) are the inputs of the ANN and \( w_{ij} \) are the weights associated with the hidden layer neurons. Besides, the number of fuzzy rules of the obtained FRBS is equal to \( 2^h \) (\( h \) is the number of hidden neurons of the ANN) and each rule has \( h \) propositions. These facts imply that obtained FRBS is very complex: \( (2^h \times h) \) propositions with inputs \( z_j \). In [2, 3] and this work an Additive Fuzzy System (AFS) is extracted from an ANN. This AFS has \( h \) rules, the antecedents of these rules use the same inputs that the ANN and each antecedent has \( n \) fuzzy propositions (\( n \) is the input number of the ANN). Thus, the FRBS has only \( (h \times n) \) propositions.

In [2, 3], The extracted AFS uses the logical operator interactive-or (defined as “symmetric sum” in [17]) in the if-part of each fuzzy rule. It is difficult to understand the action of this operator. In this paper, the obtained AFS divides the action of the \( i\text{-or} \) operator into several parts. Thus, it is easy to understand the if-part of the fuzzy rules.

In this work, AFSs with \( (h \times n) \) propositions are extracted from ANNs. These propositions are aggregated using fuzzy unions together with a compensation operator, like the fuzzy operators called uninorms [4, 20, 21, 22].

The paper is structured as follows. Initially, the relation between multi-layered feed-forward ANNs and additive fuzzy systems is introduced. Next, method to extract OR fuzzy systems is introduced. Finally, examples and conclusions are presented.
2 ANNs and AFSs

Let us consider an multi-layered feed-forward ANN with input, hidden and output layers. Let us suppose that the net has n input neurons ($x_1, ..., x_n$), h hidden neurons ($z_1, ..., z_h$), and m output neurons ($y_1, ..., y_m$). Let $\tau_j$ the bias for neuron $z_j$ and $\phi_k$ for neuron $y_k$. Let $w_{ij}$ be the weight of the connection from neuron $x_i$ to neuron $z_j$ and, $\beta_{jk}$ the weight of the connection from neuron $z_j$ to neuron $y_k$. The function the net calculates is:

$$F: \mathbb{R}^n \rightarrow \mathbb{R}^m; \quad F(x_1, ..., x_n)=(y_1, ..., y_m)$$

$$y_k = g_A \left( \sum_{j=1}^{h} (z_j \beta_{jk}) + \phi_k \right) \quad \text{with} \quad z_j = f_A \left( \sum_{i=1}^{n} (x_i w_{ij}) + \tau_j \right),$$

where $g_A$ and $f_A$ are activation functions:

- $g_A$ is usually implemented as $g_A(x) = x$.
- $f_A$ is usually a non-linear, antisymmetric, monotonic, and limited function [15]. Often, it is the sigmoid function ($\text{sigm}(x)$) or the hyperbolic tangent function ($\text{tanh}(x)$).

ANNs with sigmoid activation function can be transformed into other networks that use hyperbolic tangent function because:

$$\text{sigm}(c+d) = \text{sigm}(c) \cdot \text{sigm}(d), \quad \text{with} \quad c,d \in \mathbb{R}$$

In [2, 3], multi-layered feed-forward ANNs that use sigmoid activation function are seen as AFSs [12]. In these systems, the outputs of each rule are weighted by the activation degree of the rule and then, are added. In the obtained fuzzy system from an ANN, there is a rule $R_{jk}$ per pair of neurons (hidden, output), $(z_j, y_k)$:

$$R_{jk}: \text{If } x_1 \text{ is } A_{jk}^1 \text{ i-or } x_2 \text{ is } A_{jk}^2 \text{ i-or } ... \text{ i-or } x_n \text{ is } A_{jk}^n \text{ i-or } \tau_j \text{ is } A \text{ then } y_k = \beta_{jk},$$

where:

- The system output is the vector whose components are given by $y = \sum_{j=1}^{h} v_{jk} \beta_{jk}$, where $v_{jk}$ is the firing strength for rule $R_{jk}$ (matching degree between inputs and antecedents). The fuzzy rules $R_{jk}$ can be modified to obtain an TSK fuzzy system [3].
- There are rules “$R_{0k}$: If True then $y_k = \phi_k$”
- $A_{jk}^i$ are fuzzy sets obtained from the weights $w_{ij}$ and the fuzzy set $A$ defined by the membership function $\mu_A(x) = \text{sigm}(x)$. $A$ may be understood as “greater than approximately 2.2” because $\text{sigm}^{-1}(0.9) = 2.2$.
- The $i$-or operator is defined from the following property:

$$\text{sigm}(c+d) = \text{sigm}(c) \cdot \text{sigm}(d), \quad \text{with} \quad c,d \in \mathbb{R}$$

When we use the hyperbolic tangent activation function, a fuzzy union can be obtained in the ANN using the following property:

$$\text{tanh}(c+d) = \text{tanh}(c) \text{ OR } \text{tanh}(d), \quad \text{with} \quad c,d \in \mathbb{R}$$

3 Extracting OR fuzzy rules from ANNs

Let $z_j$ be hidden layer neuron illustrated in Fig. 1. From this neuron, the following rule can be obtained:

$$\text{If } \sum_{i=1}^{n} (x_i w_{ij}) + \tau_j \text{ then } y_k = \beta_{jk}$$

In [2, 3], multi-layered feed-forward ANNs that use sigmoid activation function are seen as AFSs [12]. In these systems, the outputs of each rule are weighted by the activation degree of the rule and then, are added. In the obtained fuzzy system from an ANN, there is a rule $R_{jk}$ per pair of neurons (hidden, output), $(z_j, y_k)$:

$$R_{jk}: \text{If } x_1 \text{ is } A_{jk}^1 \text{ i-or } x_2 \text{ is } A_{jk}^2 \text{ i-or } ... \text{ i-or } x_n \text{ is } A_{jk}^n \text{ i-or } \tau_j \text{ is } A \text{ then } y_k = \beta_{jk},$$

where:

- The system output is the vector whose components are given by $y = \sum_{j=1}^{h} v_{jk} \beta_{jk}$, where $v_{jk}$ is the firing strength for rule $R_{jk}$ (matching degree between inputs and antecedents). The fuzzy rules $R_{jk}$ can be modified to obtain an TSK fuzzy system [3].
- There are rules “$R_{0k}$: If True then $y_k = \phi_k$”
- $A_{jk}^i$ are fuzzy sets obtained from the weights $w_{ij}$ and the fuzzy set $A$ defined by the membership function $\mu_A(x) = \text{sigm}(x)$. $A$ may be understood as “greater than approximately 2.2” because $\text{sigm}^{-1}(0.9) = 2.2$.
- The $i$-or operator is defined from the following property:

$$\text{sigm}(c+d) = \text{sigm}(c) \cdot \text{sigm}(d), \quad \text{with} \quad c,d \in \mathbb{R}$$

When we use the hyperbolic tangent activation function, a fuzzy union can be obtained in the ANN using the following property:

$$\text{tanh}(c+d) = \text{tanh}(c) \text{ OR } \text{tanh}(d), \quad \text{with} \quad c,d \in \mathbb{R}$$

where $a \text{ OR } b = \frac{a \cdot b}{a \cdot b + (1 - a) \cdot (1 - b)}$ is a Hamacher family $t$-conorm [6, 9]. This $t$-conorm will be used in our extraction method.
The antecedent of the previous rule can be divided into two parts using the minus operator, as follows:

\[
\tanh \left( \sum_{i=1}^{k} (x_i \cdot w_{ij}) + \tau_j \right) = \tanh \left( \left( \sum_{i=1}^{k} (x_i \cdot w_{ij}) \right) - \left( \sum_{i=q+1}^{n} (x_i \cdot |w_{ij}|) \right) \right)
\]

we can separate the terms of the minus operator, according the following property that defines a new operator \( \ominus \):

\[
\tanh (c - d) = \tanh (c) \ominus \tanh (d), \quad c, d \in \mathbb{R}^+.
\]

With this new operator \( \ominus \), the antecedent of the rule can be expressed as follows:

\[
\tanh \left( \sum_{i=1}^{k} (x_i \cdot w_{ij}) + \tau_j \right) - \tanh \left( \sum_{i=q+1}^{n} (x_i \cdot |w_{ij}|) \right) = \tanh \left( \sum_{i=1}^{k} (x_i \cdot w_{ij}) + \tau_j \right) \ominus \tanh \left( \sum_{i=q+1}^{n} (x_i \cdot |w_{ij}|) \right)
\]

From this expression, we can use the property (1) to divide each positive part into several terms connected by a Hamacher family t-conorm:

\[
[tanh (x_j \cdot w_{ij}) \text{ OR OR OR} \ tanh(x_q \cdot w_{ij}) \text{ OR} \ tanh(\tau_j) ]
\]

\[
\ominus
\]

\[
[tanh (x_{q+1} \cdot |w_{q+i,j}|) \text{ OR OR OR} \ tanh(x_a \cdot |w_{ij}|) ]
\]

Thus, the following additive rule that use t-conorm (OR) in the if-part together with the operator \( \ominus \) can be obtained from ANNs:

**If**

\[
[tanh (x_j \cdot w_{ij}) \text{ OR ... OR} \ tanh(x_q \cdot w_{ij}) \text{ OR} \ tanh(\tau_j) ]
\]

\[
\ominus
\]

\[
[tanh (x_{q+1} \cdot |w_{q+i,j}|) \text{ OR ... OR} \ tanh(x_a \cdot |w_{ij}|) ]
\]

**then** \( y_k = \beta_{jk} \)

In order to understand this fuzzy rule, we must explain the meaning of the operator \( \ominus \) (Section 3.1) and the linguistic interpretation of the \( \tanh \) function with positive inputs (Section 3.2).

### 3.1 Intuitive meaning of the operator \( \ominus \)

**Definition 1**

The operator \( \ominus \) is defined as follows:

\[
a \ominus b = \frac{a - b}{1 - a \cdot b} = c, a, b \in [0,1) \quad \text{and} \quad c \in (-1,1).
\]

In this way, this operator fulfills the following property:

\[
\tanh (c - d) = \tanh (c) \ominus \tanh (d), \quad c, d \in \mathbb{R}^+.
\]

The meaning of this operator will be presented with a practical example together with the analysis of their properties. The practical example is:

“Let us suppose a reviewer that must evaluate a scientific paper. After an in-depth examination, he aggregates the merits of the paper into a number \( a \in [0,1) \) and the disadvantages into \( b \in [0,1) \). Starting from \( a \) and \( b \), he concludes with a number \( c \in (-1,1) \) that indicates if the paper is good \( (c > 0) \), bad \( (c < 0) \) or neuter \( (c = 0) \)”.

The behavior of the operator \( \ominus \) is equivalent to the last step carried out by the reviewer to obtain the value \( c \) from the values \( a \) and \( b \). The properties of this operator, analyzed using the example, are:

- **Property 1.** \( a \ominus a = 0.0 \).
  
  If the degree of the merits of the paper is equal to the one of the disadvantages then the conclusion is neuter.

- **Property 2:** If \( b \rightarrow 1.0 \Rightarrow a \ominus b = -1.0 \).
  
  If the paper has an important error \( (b \rightarrow 1.0) \), the conclusion is very bad.

- **Property 3:** If \( a \rightarrow 1.0 \Rightarrow a \ominus b = 1.0 \).
  
  If the paper has an very important merit \( (a \rightarrow 1.0) \), the conclusion is very good.

- **Property 4:** \( a \ominus 0 = a \).
  
  If the paper has not disadvantages then the evaluation is only influenced by the merits.

- **Property 5:** \( 0 \ominus b = -b \).
  
  If the paper has not merits then the evaluation is only influenced by the disadvantages.
• **Property 6**: If \( a > b \) \( \Rightarrow \) \( a \odot b > 0.0 \).

If the paper has a degree of merits greater than the one of disadvantages, then the evaluation is positive.

• **Property 7**: If \( a < b \) \( \Rightarrow \) \( a \odot b < 0.0 \).

If the paper has a degree of disadvantages greater than the one of merits, then the evaluation is negative.

On the other hand, this operator does not complicate the comprehension of the AFS extracted from an ANN, because:

a) Its behavior of compensation is very intuitive.

b) It is only used with two arguments in the antecedent of the extracted additive fuzzy rules.

3.2 Fuzzy interpretation of the hyperbolic tangent function

*Hyperbolic tangent* function for positive inputs is illustrated in Figure 2. This function can be understood with the linguistic expression [2]

\[ x \text{ is approximately greater than } K \equiv \text{ } x \text{ is } \approx> K \]

where \( \text{tanh} \) \((K)\) is practically 1.0. \( K = 4.0 \) can be a correct value because \( \text{tanh}(4.0) = 0.9993 \).

Hence, the expression \( \text{tan} \) \((x:w)\) with \( w \in [0,1] \) and \( w > 0.0 \) can be interpreted as

\[ x:w \text{ is approximately greater than } K \equiv \]

\[ x \text{ is approximately greater than } (K/w) \equiv \]

\[ x \text{ is } \approx> (K/w) \]

![Figure 2: Hyperbolic tangent function for positive inputs.](image)

4 Examples

_PIMA diabetes problem_[18] is a binary classification problem. This problem is used for illustrating the method that extracts AFSs with OR fuzzy rules from ANNs.

PIMA problem consists of knowing whether a person is diabetic. This problem includes 768 examples. They are divided into two sets of 576 and 192 elements that will be used as training and test examples, respectively. It is composed by eight continuous variables, normalized to \( x_i \in [0,1] \) \((i = 1, \ldots, 8)\), and one output with two possible values \((y = -1 \Rightarrow \text{No diabetes}, y = 1 \Rightarrow \text{Diabetes})\). The input variables are:

- \( x_1 \rightarrow \text{Number of pregnancies.} \)
- \( x_2 \rightarrow \text{Plasma glucose concentration a two hours in an oral glucose tolerance test.} \)
- \( x_3 \rightarrow \text{Diastolic blood pressure (mm Hg).} \)
- \( x_4 \rightarrow \text{Triceps skin fold thickness (mm).} \)
- \( x_5 \rightarrow \text{Serum insulin (µU/ml).} \)
- \( x_6 \rightarrow \text{Body mass index} \)
- \( x_7 \rightarrow \text{Diabetes pedigree function.} \)
- \( x_8 \rightarrow \text{Age in years.} \)

Multi-layered feed-forward ANNs with one hidden layer have been trained with the _Backpropagation_ algorithm [16] to solve this problem. The _hyperbolic tangent_ activation function has been used in these ANNs.

Initially, an ANN with one neuron in the hidden layer has been trained (Figure 3). This network achieves 78.47% of successes on the training examples and 78.12% of successes on the test examples. The fuzzy rules extracted from this ANN are:

\[
\text{R}_1: \text{if True then } y_1 = -0.009
\]

\[
\text{R}_2: \text{if } \begin{cases} \text{“} x_3 \text{ is } \approx> 6.03 \text{” OR “} x_5 \text{ is } \approx> 4.36 \text{” OR “} 1 \text{ is } \approx> 1.01 \text{” (=0.999)} \end{cases} \oplus \]

\[
\begin{cases} \text{“} x_4 \text{ is } \approx> 12.35 \text{” OR “} x_6 \text{ is } \approx> 1.52 \text{” OR “} x_7 \text{ is } \approx> 2.77 \text{” OR “} x_8 \text{ is } \approx> 9.02 \text{”} \end{cases}
\]

\[
\text{then } y_2 = -1.03
\]
We can observe that:

- Output of the rule $R_1$ is practically zero.
- Regarding the rule $R_2$:
  - There are two variables ($x_3$ and $x_5$) that encourage the output value “No diabetes” ($y = -1$).
  - There are seven variables ($x_1$, $x_2$, $x_4$, $x_6$, $x_7$, and $x_8$) that encourage the output value “Diabetes” ($y = 1$).
  - The fuzzy proposition (“1 is $\approx$ 1.01”) obtained from the bias value of the hidden neuron encourage the output value “No diabetes”. The degree of this proposition is very high (0.99). Therefore, the value “No diabetes” can be considered a default output value.
  - Output $y$ is obtained after aggregating the previous factors. Therefore, if we want to predict “Diabetes”, it is necessary high values for the variables $x_1$, $x_2$, $x_4$, $x_6$, $x_7$, and $x_8$. By doing this, it is compensated the degree of the propositions: “$x_3$ is $\approx$ 6.03”, “$x_5$ is $\approx$ 4.36” and “1 is $\approx$ 1.01”.

On the other hand, fuzzy propositions “$x$ is $\approx$ ($K/|w_{ij}|$)” of the rule $R_2$ with significant activation degree for inputs $x$ in [0,1] are only the propositions of the variables $x_2$, $x_6$ and $x_7$. Thus, we can suspect that these variables are very important to solve the PIMA problem.

Hence, the next experiment consists of training an ANN that only uses the input variables $x_2$, $x_6$ and $x_7$ (Figure 4). This network achieves 77.60% of successes on the training examples and 79.69% of successes on the test examples. The fuzzy rules extracted from this ANN are:

$R_1$: if True then $y_1 = -0.04$

$R_2$: if

\[
\begin{align*}
&\text{“1 is } \approx \text{ 2.66” (0.9) } \\
&\lor \text{ “} x_2 \text{ is } \approx \text{ 1.06” OR “} x_6 \text{ is } \approx \text{ 1.44” OR “} x_7 \text{ is } \approx \text{ 2.39” } \\
\end{align*}
\]

then $y_2 = -0.93$

In this new FRBS, all the fuzzy propositions have a sufficient high activation degree on the domain [0,1]. Besides, because the ANN is simple, the extracted knowledge is very intuitive:

“A person is non-diabetic (proposition “1 is $\approx$ 2.66” of rule $R_2$ that provides a default value) except when the glucose concentration ($x_2$), the body mass index ($x_6$) and the diabetes pedigree ($x_7$) have high values. In this case the person is diabetic.”

5 Conclusions

It has been presented a method to extract fuzzy rules from multi-layered feed-forward ANNs. This method has the following advantages:

- It obtains a FRBS equivalent to the ANN.
- The fuzzy rules use fuzzy unions in the antecedent.
- The input-output variables of the fuzzy system are the variables used by the ANN.
- The extracted fuzzy rules are comprehensible.
The work presented in this paper has been utilized for discovering knowledge starting from trained ANNs. It is also possible to insert knowledge into ANN before its training.

Acknowledgements

This work has been supported by the CICYT Project TIC2003-04650.

References


