Journal of Software Engineering and Applications, 2024, 17, 638-663
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

A Neuro T-Norm Fuzzy Logic Based System

Alex Tserkovny

Applied AI Services, Brookline, USA
Email: atserkovny@yahoo.com

How to cite this paper: Tserkovny, A. (2024) A Neuro T-Norm Fuzzy Logic Based System. *Journal of Software Engineering and Applications*, **17**, 638-663.
https://doi.org/10.4236/jsea.2024.178035

Received: July 11, 2024
Accepted: August 25, 2024
Published: August 28, 2024

Copyright © 2024 by author(s) and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

Abstract

In this study, we are first examining well-known approach to improve fuzzy reasoning model (*FRM*) by use of the genetic-based learning mechanism [1]. Later we propose our alternative way to build *FRM*, which has significant precision advantages and does not require any adjustment/learning. We put together neuro-fuzzy system (*NFS*) to connect the set of exemplar input feature vectors (*FV*) with associated output label (target), both represented by their membership functions (*MF*). Next unknown *FV* would be classified by getting upper value of current output *MF*. After that the fuzzy truths for all *MF* upper values are maximized and the label of the winner is considered as the class of the input *FV*. We use the knowledge in the exemplar-label pairs directly with no training. It sets up automatically and then classifies all input *FV* from the same population as the exemplar *FV*s. We show that our approach statistically is almost twice as accurate, as well-known genetic-based learning mechanism *FRM*.

Keywords

Neuro-Fuzzy System, Neural Network, Fuzzy Logic, Modus Ponnens, Modus Tollens, Fuzzy Conditional Inference

1. Introduction

Neural Network (*NN*) is regression machine that associates inputs with outputs [2]. It may represent input/output transformations, for which no models are known. A *NN* is a black box with *N* input values $X=\{x_j^q\}, j=\overline{1,N}, \ q=\overline{1,Q}$ that form a feature vector (*FV*) *X* to obtain an output vector *Z* that designates the class, identification, group, pattern, or associated output codeword of the input vector *X*. To train *NN* a set of *Q* exemplar input *FV*s is mapped to a set of output target vectors $T=\{t^q\}, q=\overline{1,Q}$, also called labels, so that each x^q maps more closely to t^q, than to another target. This allows the *NN* to make interpolations and extrapolations that map any input *X* to *Z* that best matches label *T*(*q*) for the correct index q. When trained, a *NN* is a computational machine that implements an algorithm that is specified by the input nodes,

The original backpropagation *NN*s (*BPNN*s) are trained by steepest descent on the weights that minimize the output sum-squared error E, were

$$E = \sum_{q=1,Q} \left\| z^q - t^q \right\|^2$$

Here z^q is the computed output for the input vector x^q, and t^q is the target output (label) to which x^q is supposed to map. Each z^q is a differentiable function of the weights w_{nm}, so training is done on each single weight by taking steps along the direction of steepest descent of the $\underline{E}$ via

$$w_{nm}^{i+1} = w_{nm}^{i} + \alpha \left(\frac{\partial E}{\partial w_{nm}} \right)$$

where α is the step size parameter, also called the learning rate, and i is the iteration number. The starting values of the w_{nm} are drawn randomly, usually between −0.5 and 0.5 for a cautious start. Training usually requires thousands of epochs, of which each is a set of steps to adjust each weight in $\{w_{nm}\}$ once (or sometimes more than once). However, the learning of one weight tends to unlearn the other weights, so epochs are continued until the sum-squared error is sufficiently small. Another problem of *BPNN*s is that the learned set of weights yields a local minimum, of which it has been shown that there are many [2] so that the learning is very likely to not be optimal. *BPNN*s have only a single global minimum and are thus preferable. But for most trained *NN*s there is also the problem of overtraining, by which reducing the sum-squared error to a very small value causes the noise on the input exemplars to be learned. This reduces the accuracy when other feature vectors are put through the *NN* that have different noise values.

2. Fuzzy Neural Network (FNN)

2.1. The Structure

The *FNN* in this study (**Figure 1**) is considered to be a private case of *NFS* to generate fuzzy rules and *MF*s. Note that the core of the system is multilayered network-based structure [1]. Such a system would generate both fuzzy rules and *MF*s. The source of exemplar input-output data would be described later.

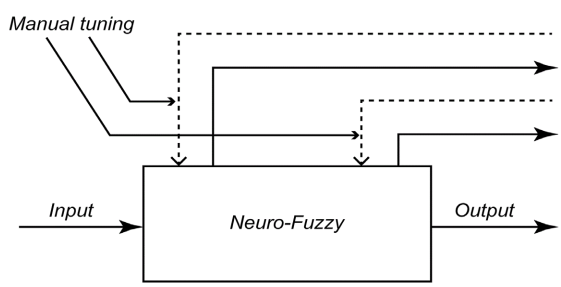

Figure 1. Neuro-fuzzy system.

A more detailed scheme of neuro-fuzzy system is depicted in **Figure 2**. For simplicity's sake we presented only two inputs X_1 and X_2 and one output Z.

The *first* layer of *neurons* simply distributes inputs of the system among neurons of the subsequent layer. The *second* layer consists of several groups of *neurons* equivalent to the number of inputs (for our case 2).

Neurons in each group represent *MF*s for fuzzy labels used as values for the input connected with this group. Output of every such *neuron* is value of membership of the input to the corresponding fuzzy level. This process is called "fuzzification" and these *neurons* are "fuzzifiers."

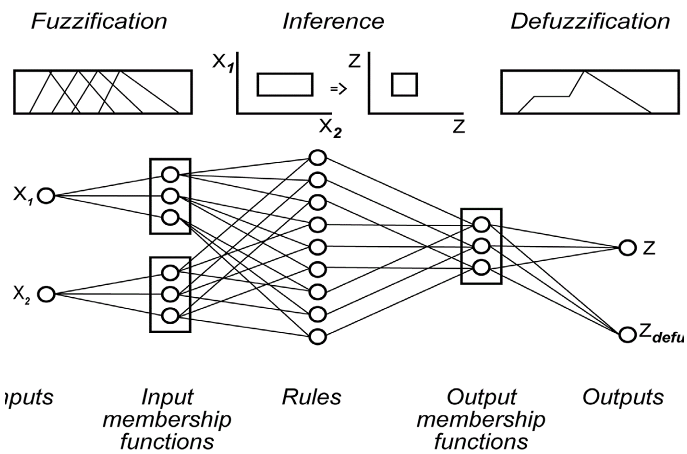

Figure 2. Detailed scheme of neuro-fuzzy system.

Neurons of the *third* layer represent fuzzy rules. Number of *neurons* in this layer can be the same as the number of rules in the logical system *IF-THEN*.

Neurons of the *fourth* layer determine *MF*s of fuzzy labels. *Neurons* of this layer perform the most complex operation, called Compositional Rule of Inference (*CRI*). Thus, the output *MF* is determined.

In the *fifth* layer the defuzzification procedure is performed. This means determination of crisp value output based on inferred fuzzy value.

Figure 2 shows a detailed structure of neuro-fuzzy system, which is like one for BPNN, mentioned in previous section, and hence allows investigation by the similar methods, but there are some differences. In *NFS* each neuron is specified not by a set of *weight*/*threshold*/universal *activation* function only, but also by complex processing unit with an individual function and set of parameters. And lastly the neurons between consecutive layers are not fully-connected unlike in case of traditional *BPNN* [1].

2.2. Fuzzy Reasoning Model

As it was mentioned above, in this study we first are examining well known [1] approach to improve *FRM* by using *genetic-based* learning mechanism. Later we propose our alternative way to build *FRM*, which has significant precision advantages and does not require any adjustment/learning.

In [1] it was stated that the selection of acceptable *MF*s is generally a subjective decision, but change in *MF*s may significantly alter the performance of the fuzzy models. It was claimed that the genetic algorithm (*GA*) allows to generate an optimal set of parameters for the fuzzy model, based either on their initial subjective selection or on a random selection.

From now on we adopt the following fuzzy conditional statements to describe a particular knowledge-based state [1]:

$$\begin{array}{c} IF\ x\ is\ A_1\ THEN\ z\ is\ B_1 \\ ALSO \\ IF\ x\ is\ A_2\ THEN\ z\ is\ B_2 \\ ALSO \\ \cdots\cdots\cdots\cdots \\ ALSO \\ IF\ x\ is\ A_q\ THEN\ z\ is\ B_q \end{array} \tag{2.1}$$

where x and z are linguistic variables, and $A_1,\cdots,A_q$ and $B_1,\cdots,A_q$ are fuzzy sets on X and Z, respectively. The fuzzy conditional statements (2.1) can be formalized in the form of the fuzzy relation $R(X,Z)$

$$R(X,Z) = ALSO(R_1, R_2, \cdots, R_i, \cdots, R_q) \tag{2.2}$$

where $ALSO$ represents a sentence connective which combines the R_i into the fuzzy relation, $R(X,Z)$ and R_i denotes the fuzzy relation between X and Z determined by the i-th fuzzy conditional statement, in which $z = B_i$ corresponds to i-th *NN*s label. The NN learning goal is to find pairs of fuzzy sets A_i and B_i, $i=\overline{1,Q}$ such that the mean square error e^2 between the fuzzy model output values and experimental output values would be the smallest. The *mean square error* e^2 is calculated by formula

$$e^2 = \frac{\sum_{i=1}^{Q}\left(z_i^* - z_i\right)^2}{\sum_{i=1}^{Q} z_i^{*2}} \tag{2.3}$$

where z_i^* is the experimental output value of the object for some current value i; z_i is the corresponding fuzzy model output value; Q is number of experiments.

In [1] the demand function $z = x \cdot \sin(1/x)$ was used to generate the set of output values z. Results are presented in **Table 1**.

Table 1. Training data.

Q	Input values	Experimental Output values
1	0.15	0.056
2	0.18	−0.120
3	0.21	−0.210
4	0.24	−0.205
5	0.27	−0.140
6	0.30	−0.057

Continued

7	0.33	0.037
8	0.36	0.128
9	0.39	0.213
10	0.42	0.290
11	0.45	0.358

Note that $x \in [0.15, 0.45]$, $z \in [-0.21, 0.358]$.

To compare our results with those from [1] we use the same linguistic descriptions of the relationship between *x* and *z* to specify the characteristics of the function:

$$
\begin{array}{l}
IF\ x = small\ THEN\ z = zero \\
ALSO \\
IF\ x = bit\ larger\ than\ small\ THEN\ z = negative\ small \\
ALSO \\
IF\ x = larger\ than\ small\ THEN\ z = negative\ large \\
ALSO \\
IF\ x = smaller\ than\ medium\ THEN\ z = negative\ large \\
ALSO \\
IF\ x = bit\ smaller\ than\ medium\ THEN\ z = negative\ medium \\
ALSO \\
IF\ x = medium\ THEN\ z = negative\ small \\
ALSO \\
IF\ x = bit\ larger\ than\ medium\ THEN\ z = zero \\
ALSO \\
IF\ x = larger\ than\ medium\ THEN\ z = positive\ small \\
ALSO \\
IF\ x = smaller\ than\ large\ THEN\ z = positive\ medium \\
ALSO \\
IF\ x = bit\ smaller\ than\ large\ THEN\ z = larger\ than\ medium \\
ALSO \\
IF\ x = large\ THEN\ z = smaller\ than\ large
\end{array}
\tag{2.4}
$$

All linguistic terms from (2.4) are defined in the following Table 2.

Table 2. Linguistic variables for input/output.

Value of variable		$x_i \in U_X, i = \overline{0,10}$ $z_j \in U_Z, i = \overline{0,7}$
X	*Z*	
small (s)	negative large (nl)	0
bit larger than small (bls)	negative medium (nm)	1
larger than small (ls)	negative small (ns)	2
smaller than medium (sm)	zero	3

Continued

bit smaller than medium (bsm)	positive small (ps)	4
Medium (m)	positive medium (pm)	5
bit larger than medium (blm)	larger than medium (lm)	6
larger than medium (lm)	smaller than large (sl)	7
smaller than large (sl)		8
bit smaller than large (bsl)		9
Large (l)		10

In [1] it was assumed that to find the crisp output value corresponding to the input value *x = 0.26* one had to successively apply the fuzzification, fuzzy logic inference mechanism and defuzzification. Experimental output value, found by formula

$$z = x \cdot \sin(1/x)\text{, was }\ z = 0.26 \cdot \sin(1/0.26) = -0.17\,.$$

In [1] membership degrees of values for both input fuzzy set, $A_i \subset U_X$, $\forall i \in [1,10]$ and output one $B_j \subset U_Z$, $\forall j \in [1,7]$, were determined by (6.1) from Appendix. From **Figure 3** we see that variable x has *11* linguistic values, whereas the variable z has *8* (see **Figure 4**) in Appendix. All linguistic values are presented in **Table 2**. The following is simulation results from [1] by (a.1):

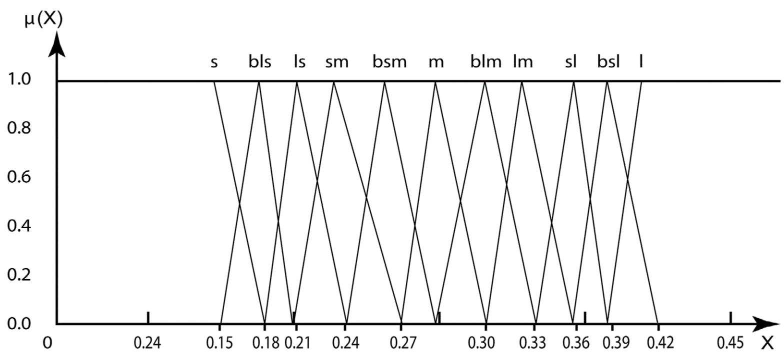

Figure 3. *MF* of fuzzy sets for input *X*.

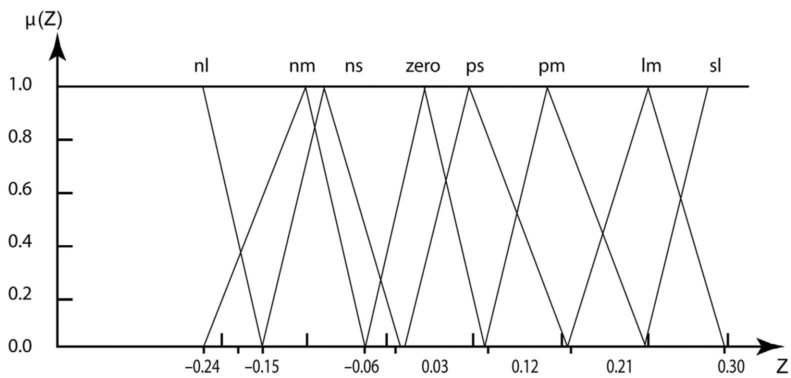

Figure 4. *MF* of fuzzy sets for output *Z*.

μ_X("0.26") = 0/0 + 0/1 + 0/2 + 0.33/3 + 0.67/4 + 0/5 + 0/6 + 0/7 + 0/8 + 0/9 + 0/10

It was shown that the knowledge-based inference mechanism was applied. The rule base (2.4), consisting of fuzzy linguistic rules, was used. Consequences of multiple (*11*) rules resulted in the fuzzy output set (see **Figure 5**), constructed on universe U_Z and bounded by the following *MF*:

μ_z("−0.17") = 0.33/0 + 0.67/1 + 0/2 + 0/3 + 0/4 + 0/5 + 0/6 + 0/7.

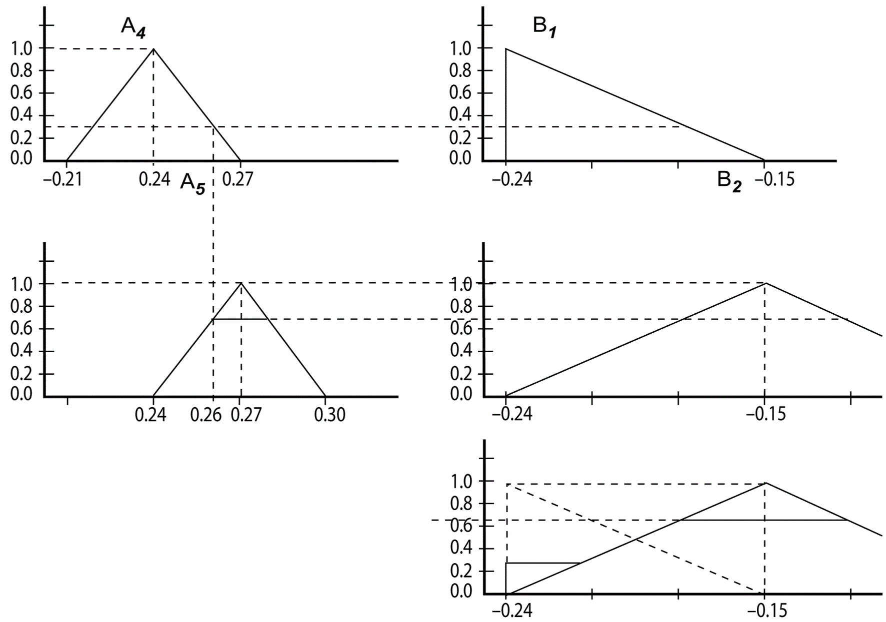

Figure 5. Geometric interpretation of inference mechanism and center of gravity method of defuzzification *X*.

Then defuzzification was applied. For this matter, the "center" of gravity defuzzification method (a.2) from Appendix was used (see **Figure 6**).

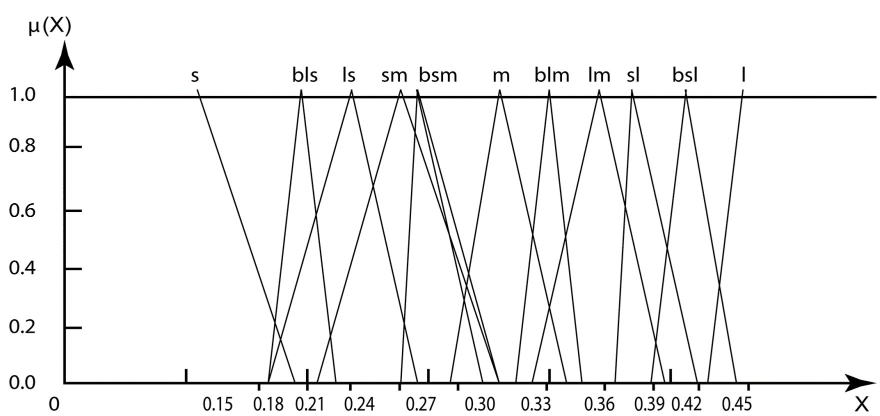

Figure 6. GA-generated improved MFs for input *X*.

Output values for given input values were calculated in the same way (see **Table 3**). Note that fuzzy rules and *MF*s were generated heuristically. In [1] it was mentioned that these rules could not provide the model precision required. To achieve the latter, it is necessary to tune appropriately the rules, as well as the shape and the center of the *MF*. To this end *GA* was used.

$$x = 0.26;\ \ y = -0.155;\ \ e^2 = 7.7854 \times 10^{-3}$$

Table 3. Comparison of models.

Q	Input values	Experimental Output values	Output values of the GA-Generated fuzzy model	Output values Of the fuzzy model	*Output of presented fuzzy model*
1	0.15	0.056	0.030	0.030	0.0334
2	0.18	−0.120	−0.091	−0.060	−0.129
3	0.21	−0.210	−0.209	−0.210	−0.21
4	0.24	−0.205	−0.210	−0.210	−0.21
5	0.27	−0.140	−0.160	−0.150	−0.13
6	0.30	−0.057	−0.060	−0.060	−0.048
7	0.33	0.037	0.027	0.030	0.033
8	0.36	0.128	0.120	0.120	0.115
9	0.39	0.213	0.196	0.210	0.196
10	0.42	0.290	0.300	0.300	0.28
11	0.45	0.358	0.360	0.360	0.358
	Mean Square Error		**0.00624**	**0.01153**	**0.00341**

2.3. Genetic-Based Learning

In [1] it was shown that 11 fuzzy sets were used in linguistic rules preconditions (see Table 2). Consequently, it was encoded 11 × 3 − 2 = 31 points. Each point $u_i, i = \overline{1,31}$ took value from a domain $D = [a_i, b_i] \subseteq U$. It was supposed that if $u_1 = 0.15$ was the value from an interval $[0.12, 0.16] \subseteq U$, then $u_2 = 0.18$ from [0.17, 0.20] etc. Then the processes of encoding and decoding were applied, they were described in both [1] and [3]. The *GA* algorithm is briefly described in Appendix.

The ***mean square errors*** for both original ($e^2 = 11.53532 \times 10^{-3}$) and *GA*-based ($e^2 = 6.24290 \times 10^{-3}$) fuzzy models are presented in Table 3.

3. T-Norm Based Approach

In contrast with above mentioned method, we are using different knowledge based one. It is built upon our unique way to fuzzification/defuzzification technique and use of t-norm based fuzzy logic [4] for logical inference, which, in general does not require additional learning. But in case of "extreme" adjustment necessity we propose a special procedure, which also based on the same fuzzy logic.

3.1. Fuzzification of Input/Output

For each $i=\overline{1,Q}$, where Q is the number of exemplar input, we represent each *FNN* input x^i as a fuzzy set, forming linguistic variable, described by a triplet of the form

$$X=\left\{\left\langle x^i,U_X,\tilde{X}\right\rangle\right\},\ x^i\in T\left(u_x\right),\ i=\overline{1,Q},$$

where $T\left(u_x\right)$ is extended term set of the linguistic variable *"Input"* from **Table 2**, $\tilde{X}$ is normal fuzzy set with correspondent *MF* $\mu_x:U_X\to\left[0,1\right]$. To normalize values of x^* we use

$$x_{norm}=\frac{x^i-x_{\min}}{x_{\max}-x_{\min}},\ i=\overline{1,Q},$$

We will use the following mapping

$$\partial:\tilde{X}\to U_X\mid u_x=Ent\left[\left(CardU_X-1\right)\times x_{norm}\right],$$

where

$$\tilde{X}=\int_{U_x}\mu_x\left(u_x\right)/u_x \tag{3.1}$$

On the other hand, to determine the estimates of the *MF* in terms of singletons from (3.1) in the form $\mu_x\left(u_{x_j}\right)/u_{x_j}\mid\forall j\in\left[0,CardU_X\right]$ we propose the following procedure.

$$\begin{aligned}&\forall j\in\left[0,CardU_X\right],\\&\mu_x\left(u_{x_j}\right)=1-\frac{1}{CardU_X-1}\times\left|j-Ent\left[\left(CardU_X-1\right)\times x_{norm}\right]\right|\end{aligned} \tag{3.2}$$

MF for an input from (3.2) is shown in **Figure 7**.

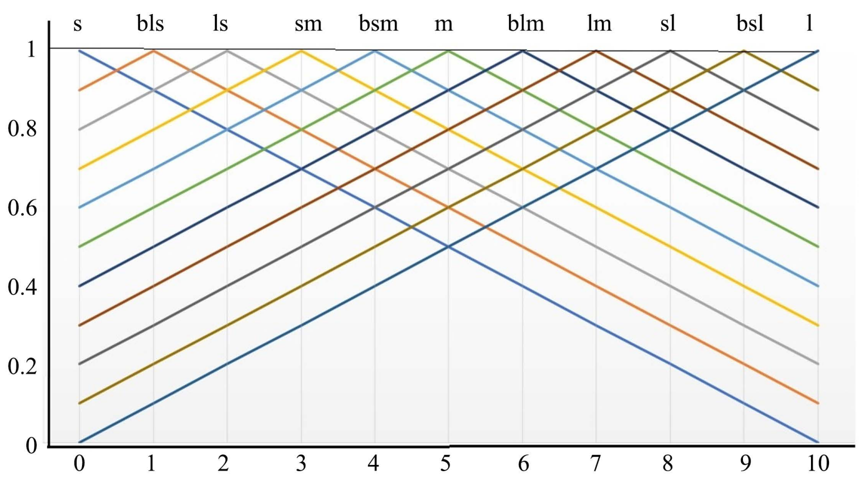

Figure 7. *MF* of fuzzy sets for *X*.

The conceptual difference between our approach to define *MF* and the one, traditionally used in fuzzy control systems, is that we define all values of a linguistic variable over entire physical scale of input/output parameters via normalization mechanism and therefore mathematically reject the notion of interval based *MF*s.

Going forward for each $i=\overline{1,Q}$, where Q is the number of exemplar output, we also represent each *FNN* output z^i as a fuzzy set, forming linguistic variable, described by a triplet of the form

$$Z=\left\{\left\langle z^i,U_Z,\tilde{Z}\right\rangle\right\},\ z^i\in T\left(u_z\right),\ i=\overline{1,Q},$$

where $T\left(u_z\right)$ is extended term set of the linguistic variable *"Output* "from Table 2, $\tilde{Z}$ is normal fuzzy set with correspondent *MF* $\mu_z:U_Z\to[0,1]$. We use the same normalization procedure

$$z_{norm}=\frac{z^i-z_{\min}}{z_{\max}-z_{\min}},\quad i=\overline{1,Q},$$

With the following mapping $\Omega:\tilde{Z}\to U_Z\,|\,u_z=Ent\left[\left(CardU_Z-1\right)\times z_{norm}\right]$, where

$$\tilde{Z}=\int_{U_Z}\mu_z\left(u_z\right)/u_z. \tag{3.3}$$

On the other hand, similarly to the previous cases, to determine the estimates of the *MF* in terms of singletons from (3.3) in the form
$\mu_z\left(u_{z_k}\right)/u_{z_k}\,|\,\forall k\in\left[0,CardU_Z\right]$ we propose the following procedure.

$$\begin{aligned}&\forall k\in\left[0,CardU_Z\right],\\&\mu_z\left(u_{zk}\right)=1-\frac{1}{CardU_Z-1}\times\left|k-Ent\left[\left(CardU_Z-1\right)\times z_{norm}\right]\right|,\end{aligned} \tag{3.4}$$

where *MF* for an output from (3.4) is shown in Figure 8.

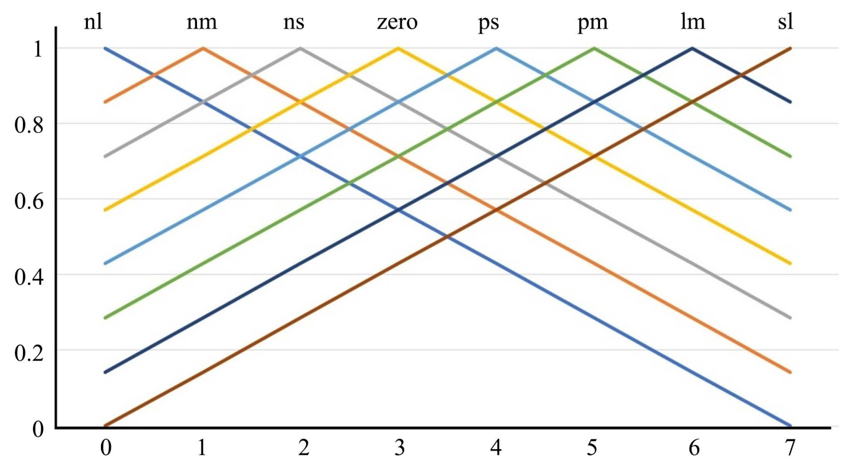

Figure 8. *MF* of fuzzy sets for *Z*.

3.2. Defuzzification of an Output

Given the fact that "Output" linguistic variable is represented by normal *MF* of the type (3.3) and for a goal of defuzzification we must find the value of index k^*, which corresponds to the following singleton value from (3.3), given (3.4)

$$\exists k^*\,|\,\mu\left(u_{z_{k^*}}\right)/u_{z_{k^*}}==1,\ \forall k\in\left[0,CardU_Z\right],$$

and the value of Output $z_{k^*} \in [z_{\max}, z_{\min}]$ would be defined as

$$z_{k^*} = k^* \times \frac{z_{\max} - z_{\min}}{CardU_Z - 1} + z_{\min} \tag{3.5}$$

3.3. Fuzzy Inference

To convert (3.1)-(3.4) into fuzzy logic-based statement and terms from Table 2 we use a *Fuzzy Conditional Inference Rule* (*FCIR*), formulated by means of "common sense" as a following conditional clause:

$$P = \text{“}IF(\tilde{X} \quad is\ X),\ THEN(\tilde{Z} \quad is\ Z)\text{”} \tag{3.6}$$

In other words, we use fuzzy conditional inference of the following type [5]:

$$\begin{array}{c} Ant\,1\text{: } If\ Input\ is\ X\ then\ Output\ is\ Z \\ Ant\,2\text{: } Input\ is\ X' \\ \text{--} \\ Cons\text{: } Output\ is\ Z'. \end{array} \tag{3.7}$$

where $X, X' \subseteq U_X$ and $Z, Z' \subseteq U_Z$.

Note that statements (3.6) and (3.7) represent "modus-ponens" syllogism. Given that we use the following type of implication [1]

$$X \to Z = \begin{cases} (1-x)\cdot z, & x > z, \\ 1, & x \le z \end{cases} \tag{3.8}$$

For practical purposes, described down below, we will use *Fuzzy Conditional Rule* (*FCR*) of the following type

$$\begin{aligned} R(A_1(x), A_2(z)) &= (X \times U_X \to U_Z \times Z) \cap (\neg X \times U_X \to U_Z \times \neg Z) \\ &= \int_{U_X \times U_Z} (\mu_x(u_x) \to \mu_z(u_z)) \wedge ((1-\mu_x(u_x)) \to (1-\mu_z(u_z))) / (u_x, u_z) \end{aligned} \tag{3.9}$$

Given (3.8) from (3.9) we are getting

$$\begin{aligned} R(A_1(x), A_2(z)) &= (\mu_x(u_x) \to \mu_z(u_z)) \wedge ((1-\mu_x(u_x)) \to (1-\mu_z(u_z))) \\ &= \begin{cases} (1-\mu_x(u_x))\cdot \mu_z(u_z), & \mu_x(u_x) < \mu_z(u_z), \\ 1, & \mu_x(u_x) = \mu_z(u_z), \\ (1-\mu_z(u_z))\cdot \mu_x(u_x), & \mu_z(u_z) < \mu_x(u_x). \end{cases} \end{aligned} \tag{3.10}$$

Given a unary relationship $R(A_1(x')) = X'$ one can obtain the consequence $R(A_2(z'))$ by *CRI* to $R(A_1(x'))$ and $R(A_1(x), A_2(z))$ of type (3.10):

$$\begin{aligned} R(A_2(z')) &= X' \circ R(A_1(x), A_2(z)) \\ &= \int_{U_X} \mu_{x'}(u_x)/u_x \circ \int_{U_X \times U_Z} (\mu_x(u_x) \to \mu_z(u_z)) \wedge ((1-\mu_x(u_x)) \to (1-\mu_z(u_z))) / (u_x, u_z) \\ &= \int_{U_Z} \bigcup_{x \in U_X} \left[\mu_{x'}(u_x) \wedge (\mu_x(u_x) \to \mu_z(u_z)) \wedge ((1-\mu_x(u_x)) \to (1-\mu_z(u_z)))\right] / u_z \end{aligned} \tag{3.11}$$

Corollary 1.

If fuzzy sets $X \subseteq U_X$ and $Z \subseteq U_Z$ are defined as (3.1) and (3.3) respectively, and are represented by *unimodal* and *normal MF*s, and also $CardU_X \ne CardU_Z$, whereas $R(A_1(x), A_2(z))$ is defined by (3.10), then the number of *singles* in matrix (3.10) is less or equal 2.

Proof:

Because of unimodality and normality of *MF*s from (3.1) and (3.3), given (3.10) and the fact that

$$\begin{aligned}&\forall j\in[0,CardU_X],\forall k\in[0,CardU_Z]\,|\,CardU_X\neq CardU_Z\\&\rightarrow\frac{1}{CardU_X-1}\times\left|j-Ent\left[(CardU_X-1)\times x_{norm}\right]\right|\\&\neq\frac{1}{CardU_Z-1}\times\left|k-Ent\left[(CardU_Z-1)\times z_{norm}\right]\right|\end{aligned}$$

the following is taking place.

1) The one *single* in a matrix is always there, because

$$\frac{1}{CardU_X-1}\times\left|j-Ent\left[(CardU_X-1)\times x_{norm}\right]\right|=0$$

and

$$\frac{1}{CardU_Z-1}\times\left|k-Ent\left[(CardU_Z-1)\times z_{norm}\right]\right|=0,$$

or

$$j^*=Ent\left[(CardU_X-1)\times x_{norm}\right]\ \text{ and }\ k^*=Ent\left[(CardU_Z-1)\times z_{norm}\right]$$

Therefore from (3.1) and (3.3)

$$\begin{aligned}&\forall j\in[0,CardU_X]\,|\,\exists!\,j^*\,|\,\mu_x\left(u_{x_{j^*}}\right)\Big/u_{x_{j^*}}=1;\\&|\,\forall k\in[0,CardU_Z]\,|\,\exists!k^*\,|\,\mu_z\left(u_{z_{k^*}}\right)\Big/u_{z_{k^*}}=1;\\&\rightarrow\mu_R\left(u_{x_{j^*}},u_{z_{k^*}}\right)\Big/(u_x,u_z)=1\end{aligned}$$

2) The *only second single* in a matrix is when

$$\frac{1}{CardU_X-1}\times\left|j-Ent\left[(CardU_X-1)\times x_{norm}\right]\right|=1$$

and

$$\frac{1}{CardU_Z-1}\times\left|k-Ent\left[(CardU_Z-1)\times z_{norm}\right]\right|=1,$$

or

$$\left|j-Ent\left[(CardU_X-1)\times x_{norm}\right]\right|=(CardU_X-1)$$

and

$$\left|k-Ent\left[(CardU_Z-1)\times z_{norm}\right]\right|=(CardU_Z-1),$$

which means $x_{norm}=1$ and $z_{norm}=1\rightarrow\exists!i^*\in[1,Q]\,|\,x^{i^*}=x_{\max}$, $z^{i^*}=z_{\max}$ and j = 0, k = 0. (Q. E. D.).

3.4. Aggregation

The aggregation (2.2) of knowledge-based situation (2.1) can be formalized in the form of the fuzzy relation $R(X,Z)$. We interpret a sentence connective *ALSO* as a fuzzy set Union

$$R(X,Z) = R_1\ OR\ R_2 \cdots OR \cdots R_i \cdots OR \cdots R_q$$

In terms of (3.9)-(3.11) we use an aggregation of the following form

$$R_{aggr}\left(A_1(x), A_2(z)\right) = \bigcup_{i=1}^{Q} R_i\left(A_1(x), A_2(z)\right) \tag{3.12}$$

3.5. Build of Neuro-Fuzzy System

We use an experimental input/output value pairs from **Table 1**.

Let us define the following in terms of neuro-fuzzy system. We are using 11 rules from (2.4). For input/output fuzzification we use (3.2) and (3.4) respectively. For *FCR* we use (3.10). For *FCIR* we use (3.11). For output defuzzification we use (3.5).

1) *Neurons* of the *second* layer (*fuzzification*) for rule 1:

μ_X(“small”) = μ_X(“0.15”) = *1.000*/0 + 0.900/1 + 0.800/2 + 0.700/3 + 0.600/4 + 0.500/5 + 0.400/6 + 0.300/7 + 0.200/8 + 0.100/9 + 0.000/10

μ_Z(“zero”) = μ_Z(“0.056”) = 0.571/0 + 0.714/1 + 0.857/2 + *1.000*/3 + 0.857/4 + 0.714/5 + 0.571/6 + 0.429/7

2) *Neurons* of the *third* layer (*FCR*) for rule 1:

$R_1(A_1(x), A_2(z))$ = (μ_X(“small”) → μ_Z(“zero”)) = (μ_X(“0.15”) → μ_Z(“0.056”)) =

$X \to Z$	0	1	2	3	4	5	6	7
0	0.000	0.000	0.000	*1.000*	0.000	0.000	0.000	0.000
1	0.057	0.071	0.086	0.000	0.086	0.071	0.057	0.043
2	0.114	0.143	0.114	0.000	0.114	0.143	0.114	0.086
3	0.171	0.200	0.100	0.000	0.100	0.200	0.171	0.129
4	0.229	0.171	0.086	0.000	0.086	0.171	0.229	0.171
5	0.214	0.143	0.071	0.000	0.071	0.143	0.214	0.214
6	0.171	0.114	0.057	0.000	0.057	0.114	0.171	0.229
7	0.129	0.086	0.043	0.000	0.043	0.086	0.129	0.171
8	0.086	0.057	0.029	0.000	0.029	0.057	0.086	0.114
9	0.043	0.029	0.014	0.000	0.014	0.029	0.043	0.057
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

3) *Neurons* of the *second* layer (*fuzzification*) for rule 2:

μ_X(“bit larger than small”) = μ_X(“0.18”) = 0.900/0 + *1.000*/1 + 0.900/2 + 0.800/3 + 0.700/4 + 0.600/5 + 0.500/6 + 0.400/7 + 0.300/8 + 0.200/9 + 0.100/10

μ_Z(“negative small”) = μ_Z(“−0.12”) = 0.857/0 + *1.000*/1 + 0.857/2 + 0.714/3 + 0.571/4 + 0.429/5 + 0.286/6 + 0.143/7

4) *Neurons* of the *third* layer (*FCR*) for rule 2:

$R_2(A_1(x), A_2(z))$ = (μ_X(“bit larger than small”) → μ_Z(“negative small”)) = (μ_X(“0.18”) → μ_Z(“−0.12”)) =

$X \rightarrow Z$	0	1	2	3	4	5	6	7
0	0.086	0.000	0.086	0.071	0.057	0.043	0.029	0.014
1	0.000	*1.000*	0.000	0.000	0.000	0.000	0.000	0.000
2	0.086	0.000	0.086	0.071	0.057	0.043	0.029	0.014
3	0.114	0.000	0.114	0.143	0.114	0.086	0.057	0.029
4	0.100	0.000	0.100	0.200	0.171	0.129	0.086	0.043
5	0.086	0.000	0.086	0.171	0.229	0.171	0.114	0.057
6	0.071	0.000	0.071	0.143	0.214	0.214	0.143	0.071
7	0.057	0.000	0.057	0.114	0.171	0.229	0.171	0.086
8	0.043	0.000	0.043	0.086	0.129	0.171	0.200	0.100
9	0.029	0.000	0.029	0.057	0.086	0.114	0.143	0.114
10	0.014	0.000	0.014	0.029	0.043	0.057	0.071	0.086

5) *Neurons* of the *second* layer (*fuzzification*) for rule 3:

μ_X(“0.21”) = 0.800/0 + 0.900/1 + *1.000*/2 + 0.900/3 + 0.800/4 + 0.700/5 + 0.600/6 + 0.500/7 + 0.400/8 + 0.300/9 + 0.200/10

μ_Z(“−0.21”) = *1.000*/0 + 0.857/1 + 0.714/2 + 0.571/3 + 0.429/4 + 0.286/5 + 0.143/6 + 0.000/7

6) *Neurons* of the *third* layer (*FCR*) for rule 3:

$R_3(A_1(x), A_2(z))$ = (μ_X(“larger than small”) → μ_Z(“negative large”)) = (μ_X(“0.21”) → μ_Z(“−0.21”)) =

$X \rightarrow Z$	0	1	2	3	4	5	6	7
0	0.000	0.114	0.143	0.114	0.086	0.057	0.029	0.000
1	0.000	0.086	0.071	0.057	0.043	0.029	0.014	0.000
2	*1.000*	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.086	0.071	0.057	0.043	0.029	0.014	0.000
4	0.000	0.114	0.143	0.114	0.086	0.057	0.029	0.000
5	0.000	0.100	0.200	0.171	0.129	0.086	0.043	0.000
6	0.000	0.086	0.171	0.229	0.171	0.114	0.057	0.000
7	0.000	0.071	0.143	0.214	0.214	0.143	0.071	0.000
8	0.000	0.057	0.114	0.171	0.229	0.171	0.086	0.000
9	0.000	0.043	0.086	0.129	0.171	0.200	0.100	0.000
10	0.000	0.029	0.057	0.086	0.114	0.143	0.114	0.000

7) *Neurons* of the *second* layer (*fuzzification*) for rule 4:

μ_X(“0.24”) = 0.700/0 + 0.800/1 + 0.900/2 + *1.000*/3 + 0.900/4 + 0.800/5 + 0.700/6 + 0.600/7 + 0.500/8 + 0.400/9 + 0.300/10

μ_Z(“−0.205”) = *1.000*/0 + 0.857/1 + 0.714/2 + 0.571/3 + 0.429/4 + 0.286/5 + 0.143/6 + 0.000/7

8) *Neurons* of the *third* layer (*FCR*) for rule 4:

$R_4(A_1(x), A_2(z))$ = μ_X(“smaller than medium”) → μ_Z(“negative large”) = (μ_X(“0.24”) → μ_Z(“−0.205”)) =

X→ *Z*	0	1	2	3	4	5	6	7
0	0.000	0.100	0.200	0.171	0.129	0.086	0.043	0.000
1	0.000	0.114	0.143	0.114	0.086	0.057	0.029	0.000
2	0.000	0.086	0.071	0.057	0.043	0.029	0.014	0.000
3	*1.000*	0.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.000	0.086	0.071	0.057	0.043	0.029	0.014	0.000
5	0.000	0.114	0.143	0.114	0.086	0.057	0.029	0.000
6	0.000	0.100	0.200	0.171	0.129	0.086	0.043	0.000
7	0.000	0.086	0.171	0.229	0.171	0.114	0.057	0.000
8	0.000	0.071	0.143	0.214	0.214	0.143	0.071	0.000
9	0.000	0.057	0.114	0.171	0.229	0.171	0.086	0.000
10	0.000	0.043	0.086	0.129	0.171	0.200	0.100	0.000

9) *Neurons* of the *second* layer (*fuzzification*) for rule 5:

μ_X(“0.27”) = 0.600/0 + 0.700/1 + 0.800/2 + 0.900/3 + *1.000*/4 + 0.900/5 + 0.800/6 + 0.700/7 + 0.600/8 + 0.500/9 + 0.400/10

μ_Z(“−0.14”) = 0.857/0 + *1.000*/1 + 0.857/2 + 0.714/3 + 0.571/4 + 0.429/5 + 0.286/6 + 0.143/7

10) *Neurons* of the *third* layer (*FCR*) for rule 5:

$R_5(A_1(x), A_2(z))$ = μ_X(“bit smaller than medium”) → μ_Z(“negative medium”) = (μ_X(“0.27”) → μ_Z(“−0.14”)) =

X→ *Z*	0	1	2	3	4	5	6	7
0	0.086	0.000	0.086	0.171	0.229	0.171	0.114	0.057
1	0.100	0.000	0.100	0.200	0.171	0.129	0.086	0.043
2	0.114	0.000	0.114	0.143	0.114	0.086	0.057	0.029
3	0.086	0.000	0.086	0.071	0.057	0.043	0.029	0.014
4	0.000	*1.000*	0.000	0.000	0.000	0.000	0.000	0.000
5	0.086	0.000	0.086	0.071	0.057	0.043	0.029	0.014
6	0.114	0.000	0.114	0.143	0.114	0.086	0.057	0.029
7	0.100	0.000	0.100	0.200	0.171	0.129	0.086	0.043
8	0.086	0.000	0.086	0.171	0.229	0.171	0.114	0.057
9	0.071	0.000	0.071	0.143	0.214	0.214	0.143	0.071
10	0.057	0.000	0.057	0.114	0.171	0.229	0.171	0.086

11) *Neurons* of the *second* layer (*fuzzification*) for rule 6:

μ_X("0.3") = 0.500/0 + 0.600/1 + 0.700/2 + 0.800/3 + 0.900/4 + *1.000*/5 + 0.900/6 + 0.800/7 + 0.700/8 + 0.600/9 + 0.500/10

μ_Z("−0.057") = 0.714/0 + 0.857/1 + *1.000*/2 + 0.857/3 + 0.714/4 + 0.571/5 + 0.429/6 + 0.286/7

12) *Neurons* of the *third* layer (*FCR*) for rule 6:

$R_6(A_1(x), A_2(z)) = (\mu_X(\text{“medium”}) \rightarrow \mu_Z(\text{“negative small”})) = (\mu_X(\text{“0.3”}) \rightarrow \mu_Z(\text{“−0.057”})) =$

$X \rightarrow Z$	0	1	2	3	4	5	6	7
0	0.143	0.071	0.000	0.071	0.143	0.214	0.214	0.143
1	0.171	0.086	0.000	0.086	0.171	0.229	0.171	0.114
2	0.200	0.100	0.000	0.100	0.200	0.171	0.129	0.086
3	0.143	0.114	0.000	0.114	0.143	0.114	0.086	0.057
4	0.071	0.086	0.000	0.086	0.071	0.057	0.043	0.029
5	0.000	0.000	*1.000*	0.000	0.000	0.000	0.000	0.000
6	0.071	0.086	0.000	0.086	0.071	0.057	0.043	0.029
7	0.143	0.114	0.000	0.114	0.143	0.114	0.086	0.057
8	0.200	0.100	0.000	0.100	0.200	0.171	0.129	0.086
9	0.171	0.086	0.000	0.086	0.171	0.229	0.171	0.114
10	0.143	0.071	0.000	0.071	0.143	0.214	0.214	0.143

13) *Neurons* of the *second* layer (*fuzzification*) for rule 7:

μ_X("0.33") = 0.400/0 + 0.500/1 + 0.600/2 + 0.700/3 + 0.800/4 + 0.900/5 + *1.000*/6 + 0.900/7 + 0.800/8 + 0.700/9 + 0.600/10

μ_Z("0.037") = 0.571/0 + 0.714/1 + 0.857/2 + *1.000*/3 + 0.857/4 + 0.714/5 + 0.571/6 + 0.429/7

14) *Neurons* of the *third* layer (*FCR*) for rule 7:

$R_7(A_1(x), A_2(z)) = (\mu_X(\text{“bit larger than medium”}) \rightarrow \mu_Z(\text{“zero”})) = (\mu_X(\text{“0.33”}) \rightarrow \mu_Z(\text{“0.037”})) =$

$X \rightarrow Z$	0	1	2	3	4	5	6	7
0	0.171	0.114	0.057	0.000	0.057	0.114	0.171	0.229
1	0.214	0.143	0.071	0.000	0.071	0.143	0.214	0.214
2	0.229	0.171	0.086	0.000	0.086	0.171	0.229	0.171
3	0.171	0.200	0.100	0.000	0.100	0.200	0.171	0.129
4	0.114	0.143	0.114	0.000	0.114	0.143	0.114	0.086
5	0.057	0.071	0.086	0.000	0.086	0.071	0.057	0.043
6	0.000	0.000	0.000	*1.000*	0.000	0.000	0.000	0.000

Continued

7	0.057	0.071	0.086	0.000	0.086	0.071	0.057	0.043
8	0.114	0.143	0.114	0.000	0.114	0.143	0.114	0.086
9	0.171	0.200	0.100	0.000	0.100	0.200	0.171	0.129
10	0.229	0.171	0.086	0.000	0.086	0.171	0.229	0.171

15) *Neurons* of the *second* layer (*fuzzification*) for rule 8:

μ_X(“0.36”) = 0.300/0 + 0.400/1 + 0.500/2 + 0.600/3 + 0.700/4 + 0.800/5 + 0.900/6 + *1.000*/7 + 0.900/8 + 0.800/9 + 0.700/10

μ_Z(“0.128”) = 0.429/0 + 0.571/1 + 0.714/2 + 0.857/3 + *1.000*/4 + 0.857/5 + 0.714/6 + 0.571/7

16) *Neurons* of the *third* layer (*FCR*) for rule 8:

$R_8(A_1(x), A_2(z))$ = (μ_X(“larger than medium”) → μ_Z(“positive small”)) = (μ_X(“0.36”) → μ_Z(“0.128”)) =

$X \to Z$	0	1	2	3	4	5	6	7
0	0.171	0.129	0.086	0.043	0.000	0.043	0.086	0.129
1	0.229	0.171	0.114	0.057	0.000	0.057	0.114	0.171
2	0.214	0.214	0.143	0.071	0.000	0.071	0.143	0.214
3	0.171	0.229	0.171	0.086	0.000	0.086	0.171	0.229
4	0.129	0.171	0.200	0.100	0.000	0.100	0.200	0.171
5	0.086	0.114	0.143	0.114	0.000	0.114	0.143	0.114
6	0.043	0.057	0.071	0.086	0.000	0.086	0.071	0.057
7	0.000	0.000	0.000	0.000	*1.000*	0.000	0.000	0.000
8	0.043	0.057	0.071	0.086	0.000	0.086	0.071	0.057
9	0.086	0.114	0.143	0.114	0.000	0.114	0.143	0.114
10	0.129	0.171	0.200	0.100	0.000	0.100	0.200	0.171

17) *Neurons* of the *second* layer (*fuzzification*) for rule 9:

μ_X(“0.39”) = 0.200/0 + 0.300/1 + 0.400/2 + 0.500/3 + 0.600/4 + 0.700/5 + 0.800/6 + 0.900/7 + *1.000*/8 + 0.900/9 + 0.800/10

μ_Z(“0.213”) = 0.286/0 + 0.429/1 + 0.571/2 + 0.714/3 + 0.857/4 + *1.000*/5 + 0.857/6 + 0.714/7

18) *Neurons* of the *third* layer (*FCR*) for rule 9:

$R_9(A_1(x), A_2(z))$ = (μ_X(“smaller than large”) → μ_Z(“positive medium”)) = (μ_X(“0.39”) → μ_Z(“0.213”)) =

$X \to Z$	0	1	2	3	4	5	6	7
0	0.143	0.114	0.086	0.057	0.029	0.000	0.029	0.057

Continued

1	0.200	0.171	0.129	0.086	0.043	0.000	0.043	0.086
2	0.171	0.229	0.171	0.114	0.057	0.000	0.057	0.114
3	0.143	0.214	0.214	0.143	0.071	0.000	0.071	0.143
4	0.114	0.171	0.229	0.171	0.086	0.000	0.086	0.171
5	0.086	0.129	0.171	0.200	0.100	0.000	0.100	0.200
6	0.057	0.086	0.114	0.143	0.114	0.000	0.114	0.143
7	0.029	0.043	0.057	0.071	0.086	0.000	0.086	0.071
8	0.000	0.000	0.000	0.000	0.000	*1.000*	0.000	0.000
9	0.029	0.043	0.057	0.071	0.086	0.000	0.086	0.071
10	0.057	0.086	0.114	0.143	0.114	0.000	0.114	0.143

19) *Neurons* of the *second* layer (*fuzzification*) for rule 10:

μ_X(“0.42”) = 0.100/0 + 0.200/1 + 0.300/2 + 0.400/3 + 0.500/4 + 0.600/5 + 0.700/6 + 0.800/7 + 0.900/8 + *1.000*/9 + 0.900/10

μ_Z(“0.29”) = 0.143/0 + 0.286/1 + 0.429/2 + 0.571/3 + 0.714/4 + 0.857/5 + *1.000*/6 + 0.857/7

20) *Neurons* of the *third* layer (*FCR*) for rule 10:

$R_{10}(A_1(x), A_2(z))$ = (μ_X(“bit smaller than large”) → μ_Z(“larger than medium”)) = (μ_X(“0.42”) → μ_Z(“0.29”)) =

$X \to Z$	0	1	2	3	4	5	6	7
0	0.086	0.071	0.057	0.043	0.029	0.014	0.000	0.014
1	0.114	0.143	0.114	0.086	0.057	0.029	0.000	0.029
2	0.100	0.200	0.171	0.129	0.086	0.043	0.000	0.043
3	0.086	0.171	0.229	0.171	0.114	0.057	0.000	0.057
4	0.071	0.143	0.214	0.214	0.143	0.071	0.000	0.071
5	0.057	0.114	0.171	0.229	0.171	0.086	0.000	0.086
6	0.043	0.086	0.129	0.171	0.200	0.100	0.000	0.100
7	0.029	0.057	0.086	0.114	0.143	0.114	0.000	0.114
8	0.014	0.029	0.043	0.057	0.071	0.086	0.000	0.086
9	0.000	0.000	0.000	0.000	0.000	0.000	*1.000*	0.000
10	0.014	0.029	0.043	0.057	0.071	0.086	0.000	0.086

21) *Neurons* of the *second* layer (*fuzzification*) for rule 11:

μ_X(“0.45”) = 0.000/0 + 0.100/1 + 0.200/2 + 0.300/3 + 0.400/4 + 0.500/5 + 0.600/6 + 0.700/7 + 0.800/8 + 0.900/9 + *1.000*/10

μ_Z(“0.358”) = 0.000/0 + 0.143/1 + 0.286/2 + 0.429/3 + 0.571/4 + 0.714/5 + 0.857/6 + *1.000*/7

22) *Neurons* of the *third* layer (*FCR*) for rule 11:

$R_{11}(A_1(x), A_2(z)) = (\mu_X(\text{“large”}) \to \mu_Z(\text{“smaller than large”})) = (\mu_X(\text{“0.45”}) \to \mu_Z(\text{“0.358”})) =$

$X \to Z$	0	1	2	3	4	5	6	7
0	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.086	0.071	0.057	0.043	0.029	0.014	0.000
2	0.000	0.114	0.143	0.114	0.086	0.057	0.029	0.000
3	0.000	0.100	0.200	0.171	0.129	0.086	0.043	0.000
4	0.000	0.086	0.171	0.229	0.171	0.114	0.057	0.000
5	0.000	0.071	0.143	0.214	0.214	0.143	0.071	0.000
6	0.000	0.057	0.114	0.171	0.229	0.171	0.086	0.000
7	0.000	0.043	0.086	0.129	0.171	0.200	0.100	0.000
8	0.000	0.029	0.057	0.086	0.114	0.143	0.114	0.000
9	0.000	0.014	0.029	0.043	0.057	0.071	0.086	0.000
10	0.000	0.000	0.000	0.000	0.000	0.000	0.000	*1.000*

23) *Neurons* of the *third* layer (*FCR*) aggregation:

$$R_{aggr}\left(A_1(x), A_2(z)\right) = \bigcup_{k=1}^{11} R_k\left(A_1(x), A_2(z)\right) =$$

$X \to Z$	0	1	2	3	4	5	6	7
0	*1.000*	0.129	0.200	*1.000*	0.229	0.214	0.214	0.229
1	0.229	1.000	0.143	0.200	0.171	0.229	0.214	0.214
2	*1.000*	0.229	0.171	0.143	0.200	0.171	0.229	0.214
3	*1.000*	0.229	0.229	0.171	0.143	0.200	0.171	0.229
4	0.229	*1.000*	0.229	0.229	0.171	0.171	0.229	0.171
5	0.214	0.143	*1.000*	0.229	0.229	0.171	0.214	0.214
6	0.171	0.114	0.200	*1.000*	0.229	0.214	0.171	0.229
7	0.143	0.114	0.171	0.229	*1.000*	0.229	0.171	0.171
8	0.200	0.143	0.143	0.214	0.229	*1.000*	0.200	0.114
9	0.171	0.200	0.143	0.171	0.229	0.229	*1.000*	0.129
10	0.229	0.171	0.200	0.143	0.171	0.229	0.229	*1.000*

24) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 1:

$\mu_Z(\text{“zero”}) = \mu_X(\text{“small”}) \circ R_{aggr}(A_1(x), A_2(z)) = 1.000/0 + 0.900/1 + 0.500/2 + \mathit{1.000}/3 + 0.300/4 + 0.229/5 + 0.229/6 + 0.229/7$

25) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 1:

Defuzzification of $\mu_Z(\text{“zero”}) \Rightarrow 0.03342857142857139$

26) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 2:

μ_Z("negative medium") = μ_X("bit larger than small") ∘ $R_{aggr}(A_1(x), A_2(z))$ = 0.900/0 + *1.000*/1 + 0.600/2 + 0.900/3 + 0.400/4 + 0.300/5 + 0.229/6 + 0.229/7

27) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 2:

Defuzzification of μ_Z("negative medium") ⇒ −0.12885714285714284

28) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 3:

μ_Z("negative large") = μ_X("larger than small") ∘ $R_{aggr}(A_1(x), A_2(z))$ = *1.000*/0 + 0.900/1 + 0.700/2 + 0.800/3 + 0.500/4 + 0.400/5 + 0.300/6 + 0.229/7

29) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 3:

Defuzzification of μ_Z("negative large") ⇒ −0.21

30) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 4:

μ_Z("negative large") = μ_X("smaller than medium") ∘ $R_{aggr}(A_1(x), A_2(z))$ = *1.000*/0 + 0.900/1 + 0.800/2 + 0.700/3 + 0.600/4 + 0.500/5 + 0.400/6 + 0.300/7

31) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 4:

Defuzzification of μ_Z("negative large") ⇒ −0.21

32) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 5:

μ_Z("negative medium") = μ_X("bit smaller than medium") ∘ $R(A_1(x), A_2(z))$ = 0.900/0 + *1.000*/1 + 0.900/2 + 0.800/3 + 0.700/4 + 0.600/5 + 0.500/6 + 0.400/7

33) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 5:

Defuzzification of μ_Z("*negative medium*") ⇒ −0.12885714285714284

34) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 6:

μ_Z("negative small") = μ_X("medium") ∘ $R_{aggr}(A_1(x), A_2(z))$ = 0.800/0 + 0.900/1 + *1.000*/2 + 0.900/3 + 0.800/4 + 0.700/5 + 0.600/6 + 0.500/7

35) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 6:

Defuzzification of μ_Z("negative small") ⇒ −0.04771428571428571

36) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 7:

μ_Z("*zero*") = μ_X("bit larger than medium") ∘ $R_{aggr}(A_1(x), A_2(z))$ = 0.700/0 + 0.800/1 + 0.900/2 + *1.000*/3 + 0.900/4 + 0.800/5 + 0.700/6 + 0.600/7

37) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 7:

Defuzzification of μ_Z("zero") ⇒ 0.03342857142857139

38) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 8:

μ_Z("positive small") = μ_X("larger than medium") ∘ $R_{aggr}(A_1(x), A_2(z))$ = 0.600/0 + 0.700/1 + 0.800/2 + 0.900/3 + *1.000*/4 + 0.900/5 + 0.800/6 + 0.700/7

39) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 8:

Defuzzification of μ_Z("*positive small*") ⇒ 0.11457142857142857

40) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 9:

μ_Z("positive medium") = μ_X("smaller than large") ∘ $R_{aggr}(A_1(x), A_2(z))$ = 0.500/0 + 0.600/1 + 0.700/2 + 0.800/3 + 0.900/4 + *1.000*/5 + 0.900/6 + 0.800/7

41) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 9:

Defuzzification of μ_Z("positive medium") ⇒ 0.1957142857142857

42) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 10:

μ_Z("larger than medium") = μ_X("bit smaller than large") ∘ $R_{aggr}(A_1(x), A_2(z))$ = 0.400/0 + 0.500/1 + 0.600/2 + 0.700/3 + 0.800/4 + 0.900/5 + *1.000*/6 + 0.900/7

43) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 10:

Defuzzification of μ_Z("larger than medium") $\Rightarrow$ 0.2768571428571428

44) *Neurons* of the *fourth* layer (*FCIR*) composition for rule 11:

μ_Z("smaller than large") = μ_X("large") $\circ$ $R_{aggr}(A_1(x), A_2(z))$ = 0.300/0 + 0.400/1 + 0.500/2 + 0.600/3 + 0.700/4 + 0.800/5 + 0.900/6 + *1.000*/7

45) *Neurons* of the *fifth* layer (*Defuzzification*) for output of rule 11:

Defuzzification of μ_Z("smaller than large") $\Rightarrow$ 0.358.

The ***mean square error*** for fuzzy model based on our t-norm approach $e^2 = 3.41322 \times 10^{-3}$ is shown in Table 3. This result statistically is *almost twice as accurate*, as GA-Generated fuzzy model.

3.6. Binary Rules Adjustment by New Label

In real world of *NN* based systems a value of their input/output pairs might be significantly changed in accordance with a set of a new requirements/capabilities. It could be a situation of a new *label/class* introduction. The latter means that *aggregated FCR* matrix of a system $R_{aggr}\left(A_1(x), A_2(z)\right)$ must be modified, based on an *additional label*, never used originally. We presume that the value of a *new label* could situate outside of the scale of normalized output values $z_{\text{norm}} \in \left[z_{\max}, z_{\min}\right]$, used initially. At this case one must do the following.

1) Expand *original scale* or re-scale both labels/potential input pairs like that

$$z'_{\text{norm}} \in \left[z_{\max} + \Delta z, z_{\min} - \Delta z\right], \quad x'_{\text{norm}} \in \left[x_{\max} + \Delta x, x_{\min} - \Delta x\right], \tag{3.13}$$

where

$$\Delta z = \begin{cases} \left|z_{\text{label}} - z_{\max}\right|, z_{\max} < z_{\text{label}} \\ \left|z_{\min} - z_{\text{label}}\right|, z_{\text{label}} < z_{\min} \end{cases} \tag{3.14}$$

On practice the value of $\Delta z = \Delta z + \varepsilon$, when ε is defined empirically. In general terms could be the following linear function $\varepsilon = f\left(\Delta z\right)$.

2) Find the input value, which corresponds to the new label/class.

For this matter we would use Generalized Modus Tollens [6] mechanism, the scheme of which is the following

$$\begin{array}{c} Ant\,1\text{: } IF\ x\ is\ A\ THEN\ z\ is\ B \\ Ant\,2\text{: } z\ is\ B' \\ \text{---} \\ \text{Cons: } x\ is\ A'. \end{array} \tag{3.15}$$

The *most important* thing to mention is that in (3.15) *Ant* 1, is represented by *aggregated FCR* matrix of a system $R_{aggr}\left(A_1(x), A_2(z)\right)$.

In terms of *FCR*, given a unary relationship $R\left(A_2(z')\right) = B'$ one can obtain the consequence $R\left(A_1(x')\right)$ by *CRI* by applying it to $R\left(A_2(z')\right)$ and $R_{aggr}\left(A_1(x), A_2(z)\right)$ of type (3.10):

$$\begin{aligned} R\left(A_1(x')\right) &= R\left(A_2(z')\right) \circ R_{aggr}\left(A_1(x), A_2(z)\right) \\ &= \int_{U_Z} \mu_{z'}(u_z)/u_z \circ \int_{U_X \times U_Z} \left(\mu_x(u_x) \to \mu_z(u_z)\right) \wedge \left(\left(1-\mu_x(u_x)\right) \to \left(1-\mu_z(u_z)\right)\right) \Big/ (u_x, u_z) \\ &= \int_{U_X} \bigcup_{z \in U_Z} \left[\mu_{z'}(u_z) \wedge \left(\mu_x(u_x) \to \mu_z(u_z)\right) \wedge \left(\left(1-\mu_x(u_x)\right) \to \left(1-\mu_z(u_z)\right)\right)\right] \Big/ u_x \end{aligned} \tag{3.16}$$

3) Based on *CRI* (3.16) add *neuron* of the *third* layer (*FCR*) for new rule:

$$\begin{aligned} R_{new}\left(A_1(x),A_2(z)\right) &= \left(X'\times U_X \to U_Z\times Z'\right)\cap\left(\neg X'\times U_X \to U_Z\times\neg Z'\right) \\ &= \int_{U_X\times U_Z}\left(\mu_{x'}(u_x)\to\mu_{z'}(u_z)\right)\wedge\left(\left(1-\mu_{x'}(u_x)\right)\to\left(1-\mu_{z'}(u_z)\right)\right)\Big/\left(u_x,u_z\right) \end{aligned} \quad (3.17)$$

4) Repeat an aggregation of *neurons* of the *third* layer (*FCR*) by using (3.17) and by previously *aggregated FCR* matrix of a system $R_{aggr}\left(A_1(x),A_2(z)\right)$.

$$R'_{aggr}\left(A_1(x),A_2(z)\right)=R_{new}\left(A_1(x),A_2(z)\right)\cup R_{aggr}\left(A_1(x),A_2(z)\right) \quad (3.18)$$

This way we incorporated new knowledge into our system.

3.7. The Instance of Binary Rules Adjustment

1) Suppose we have the *new label* $z'=0.37$ and let $\Delta x=0.05$, $\Delta z=0.02$. Therefore expand (re-scale) both labels/potential input pairs like that $x'\in[0.1,0.5]$, $z'\in[-0.23,0.378]$.

2) The *fuzzified* value for $z'=0.37$ from (3.13) and (3.4) is $\forall k\in\left[0,CardU_Z\right]$,

$$\mu_{z'}\left(u_{z_k}\right)=1-\frac{1}{CardU_Z-1}\times\left|k-Ent\left[\left(CardU_Z-1\right)\times bz'_{norm}\right]\right|, \text{ i.e.}$$

μ_z(“0.37”) = 0.000/0 + 0.143/1 + 0.286/2 + 0.429/3 + 0.571/4 + 0.714/5 + 0.857/6 + 1.000/7

3) After application of *Generalized Modus Tollens* (3.15) and (3.16), *i.e.*

$$\begin{aligned} R\left(A_1(x')\right) &= Rb\left(A_2(z')\right)\circ R_{aggr}\left(A_1(x),A_2(z)\right) \\ &= \int_{U_Z}\mu_{z'}(u_z)/u_z\circ\int_{U_X\times U_Z}\left(\mu_x(u_x)\to\mu_z(u_z)\right) \\ &\quad\wedge\left(\left(1-\mu_x(u_x)\right)\to\left(1-\mu_z(u_z)\right)\right)\Big/\left(u_x,u_z\right) \end{aligned}$$

we are getting

μ_x(“large”) = 0.429/0 + 0.229/1 + 0.229/2 + 0.229/3 + 0.229/4 + 0.286/5 + 0.429/6 + 0.571/7 + 0.714/8 + 0.857/9 + 1.000/10

4) Defuzzification of μ_x(“large”) $\Rightarrow$ 0.5.

5) From (3.17) we build binary matrix for the new rule

$$R_{new}\left(A_1(x),A_2(z)\right)=\left(X'\times U_X\to U_Z\times Z'\right)\cap\left(\neg X'\times U_X\to U_Z\times\neg Z'\right)=$$

0.000	0.082	0.163	*1.000*	0.184	0.122	0.061	0.000
0.000	0.110	0.163	0.131	0.098	0.065	0.033	0.000
0.000	0.110	0.163	0.131	0.098	0.065	0.033	0.000
0.000	0.110	0.163	0.131	0.098	0.065	0.033	0.000
0.000	0.110	0.163	0.131	0.098	0.065	0.033	0.000
0.000	0.102	*1.000*	0.163	0.122	0.082	0.041	0.000
0.000	0.082	0.163	*1.000*	0.184	0.122	0.061	0.000
0.000	0.061	0.122	0.184	*1.000*	0.163	0.082	0.000
0.000	0.041	0.082	0.122	0.163	*1.000*	0.102	0.000
0.000	0.020	0.041	0.061	0.082	0.102	*1.000*	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	*1.000*

6) Repeat an aggregation of *neurons* of the *third* layer by using (3.18)

$$R'_{aggr}\left(A_1(x), A_2(z)\right) =$$

1.000	0.129	0.200	*1.000*	0.229	0.214	0.214	0.229
0.229	*1.000*	0.163	0.200	0.171	0.229	0.214	0.214
1.000	0.229	0.171	0.143	0.200	0.171	0.229	0.214
1.000	0.229	0.229	0.171	0.143	0.200	0.171	0.229
0.229	*1.000*	0.229	0.229	0.171	0.171	0.229	0.171
0.214	0.143	*1.000*	0.229	0.229	0.171	0.214	0.214
0.171	0.114	0.200	*1.000*	0.229	0.214	0.171	0.229
0.143	0.114	0.171	0.229	*1.000*	0.229	0.171	0.171
0.200	0.143	0.143	0.214	0.229	*1.000*	0.200	0.114
0.171	0.200	0.143	0.171	0.229	0.229	*1.000*	0.129
0.229	0.171	0.200	0.143	0.171	0.229	0.229	*1.000*

7) Unit test $R'_{aggr}\left(A_1(x), A_2(z)\right)$ by using μ_x(“**0.5**”). For this matter apply fuzzification (3.2) and get

$R(A_1(x))$ = μ_x(“**0.5**”) = 0.000/0 + 0.100/1 + 0.200/2 + 0.300/3 + 0.400/4 + 0.500/5 + 0.600/6 + 0.700/7 + 0.800/8 + 0.900/9 + 1.000/10.

Obtain the consequence $R\left(A_2(z)\right)$ by *CRI* to $R\left(A_1(x)\right)$ and $R'_{aggr}\left(A_1(x), A_2(z)\right)$ of type (3.10):

$$R\left(A_2(z)\right) = R\left(A_1(x)\right) \circ R'_{age^2gr}\left(A_1(x), A_2(z)\right)$$

and get μ_z(“smaller than large”) = 0.300/0 + 0.400/1 + 0.500/2 + 0.600/3 + 0.700/4 + 0.800/5 + 0.900/6 + 1.000/7.

Defuzzification of μ_z(“smaller than large”) $\Rightarrow$ **0.378.** The *mean square error* for the case $e^2 = 4.675 \times 10^{-4}$, which is *extremely* precise result, confirming the legitimacy of the approach.

4. Conclusion

In this study, we first examined well-known [1] *FRM* with genetic-based learning mechanism. We proposed an alternative way to build *FRM*, which does not require any adjustment/learning. We have shown that our approach is statistically almost twice as accurate, as the well-known *FRM*, which uses a genetic-based learning mechanism. We have introduced the label-driven binary relationship matrix adjustment technique.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Aliev, R.A., Fazlollahi, B. and Aliev, R.R. (2004) Soft Computing and Its Application in Business and Economics. Physica-Verlag, Springer. https://doi.org/10.1007/978-3-540-44429-9

[2] Looney, C.G. and Dascalu, S. (2007) A Simple Fuzzy Neural Network. https://www.cse.unr.edu/~looney/cs773b/fuzzyNNbk.pdf

[3] Aliev, R.A., Fazlollahi, B. and Vahidov, R.M. (2001) Genetic Algorithm-Based Learning of Fuzzy Neural Networks. Part 1: Feed-Forward Fuzzy Neural Networks. *Fuzzy Sets and Systems*, **118**, 351-358. https://doi.org/10.1016/s0165-0114(98)00461-8

[4] Tserkovny, A. (2017) A T-Norm Fuzzy Logic for Approximate Reasoning. *Journal of Software Engineering and Applications*, **10**, 639-662. https://doi.org/10.4236/jsea.2017.107035

[5] Fukami, S., Mizumoto, M. and Tanaka, K. (1980) Some Considerations on Fuzzy Conditional Inference. *Fuzzy Sets and Systems*, **4**, 243-273. https://doi.org/10.1016/0165-0114(80)90014-7

[6] Tserkovny, A. (2017) A Fuzzy Logic Based Resolution Principal for Approximate Reasoning. *Journal of Software Engineering and Applications*, **10**, 793-823. https://doi.org/10.4236/jsea.2017.1010045

Appendix

The interval based *MF*, used in [1]

$$\mu\left(x,a_{i1},a_{i2},a_{i3}\right)=\begin{cases}0, & \text{if } x\ll a_{i1}\\ \dfrac{x-a_{i1}}{a_{i2}-a_{i1}}, & \text{if } a_{i1}\ll x\ll a_{i2}\\ \dfrac{a_{i3}-x}{a_{i3}-a_{i2}}, & \text{if } a_{i2}\ll x\ll a_{i3}\\ 0, & \text{if } x\gg a_{i3}\end{cases} \tag{a.1}$$

where a_{i1},a_{i2},a_{i3} are tuning parameters for *i*-th fuzzy subset

$$a_{i1}^{n}=\left(a_{i1}+\delta_i\right)-\tau_i,$$

$$a_{i2}^{n}=\left(a_{i2}+\delta_i\right),$$

$$a_{i3}^{n}=\left(a_{i3}+\delta_i\right)+\tau_i,$$

where δ_i,τ_i are some tuning coefficients. The parameter δ_i shifts *MF* to the left or to the right. The parameter τ_i allows changing the shape of *MF*.

$$z_c=\frac{\int_{-0.24}^{0.39} z\cdot\mu_z \mathrm{d}z}{\int_{-0.24}^{0.39}\mu_z \mathrm{d}z}=-0.155 \tag{a.2}$$

The summary of the referenced fuzzy model, proposed in [1] is the following.

1) *Define fussy sets for input* $A_i\subseteq U_X$, $\forall i\in[1,l]$ *and output one* $B_j\subseteq U_Z$, $\forall j\in[1,p]$

2) *Determine linguistic* (*fuzzy*) *rules.*

3) *Implement the justification process. During the fuzzification the values of input variable are transformed by using stored MFs to produce fuzzy input values.*

4) *Activate knowledge-based fuzzy logic inference mechanism. Generate fuzzy output value.*

5) *Execute defuzzification process. It results in crisp value of the output fuzzy value.*

6) *Calculate by Formula* (2.3) *the mean square error* e^2 *for each input value.*

7) *If e is less than the given precision, go to step* 17*.*

8) *Start the GA work* $t=1$*.*

9) *Create the initial population.*

10) *Evaluate* $G(t)$*. This step also consists of fuzzification, inference, defuzzification, which precede calculation of the mean square error for each chromosome* c_i, $i=\overline{1,ps}$*. Besides, minimum square error is stored in memory.*

11) *If some termination conditions are met, go to step* 15*.*

12) *Produce new generation* $G(t+1)$ *from* $G(t)$*. Then crossover and mutation are applied.*

13) *Evaluate* $G(t+1)$*.*

14) *Return to step* 11*.*

15) *Terminate GA's work.*

16) *Find the smallest one among all minimum errors stored in memory. Select the fuzzy set* A_i, $i=\overline{1,n}$ *and crisp output value, by which the smallest mean square error obtained.*

17) *End.*