Article

A Symmetrical Fuzzy Neural Network Regression Method Coordinating Structure and Parameter Identifications for Regression

Ke Zhang, Wenning Hao *, Xiaohan Yu * and Tianhao Shao

Command & Control Engineering College, Army Engineering University of PLA, Nanjing 210007, China
* Correspondence: hwnbox@aeu.edu.cn.com (W.H.); yuxh@aeu.edu.cn (X.Y.)

Abstract: Fuzzy neural networks have both the interpretability of fuzzy systems and the self-learning ability of neural networks, but they will face the challenge of "rule explosion" when dealing with high-dimensional data. Moreover, the structure and parameter identifications of models are generally performed in two stages, and this always attends to one thing and loses another in terms of interpretability and predictive performance. In this paper, a fuzzy neural network regression method (FNNR) that coordinates structure identification and parameter identification is proposed. To alleviate the problem of rule explosion, the structure identification and parameter identification are coordinated in the training process, and the numbers of fuzzy rules and fuzzy partitions are effectively limited, while the parameters of fuzzy rules are optimized. The symmetrical architecture of the FNNR is designed for automatic structure identification. An alternate training strategy is adopted by treating discrete and continuous parameters differently, and thus the convergence efficiency of the algorithm is improved. To enhance interpretability, regularized terms are designed from fuzzy rule level and fuzzy partition level to guide the model to learn fuzzy rules with simple structures and clear semantics. The experimental results show that the proposed method has both a compact structure and high precision.

Keywords: fuzzy neural networks; TSK fuzzy systems; regression; interpretability; symmetrical architecture

Citation: Zhang, K.; Hao, W; Yu, X; Shao, T. A Symmetrical Fuzzy Neural Network Regression Method Coordinating Structure and Parameter Identifications for Regression. *Symmetry* **2023**, *15*, 1711. https://doi.org/10.3390/sym15091711

Academic Editor: Alice Miller

Received: 27 July 2023
Revised: 21 August 2023
Accepted: 29 August 2023
Published: 6 September 2023

1. Introduction

Fuzzy neural networks (FNNs) [1] have both interpretability and a self-learning ability, achieved by combining fuzzy systems [2,3] and neural networks, and they are important parts of the prevailing eXplainable Artificial Intelligence (XAI) field [4]. The FNN can learn a set of fuzzy IF-THEN rules with appropriate linguistic labels from data, so that it is easy to understand the decision-making process of the model. In addition, the FNN can improve its performance via an iterative algorithm. With a powerful capability for knowledge representation and learning, FNNs have been widely applied to related fields [5–9].

The identifications of FNNs can be divided into two parts: structure identification and parameter identification [10]. Structure identification refers to finding appropriate fuzzy partitions for the input space and determining the number of fuzzy rules. Parameter identification means determining the parameters of the antecedents and consequents of the fuzzy rules. In structure identification, the key work is to determine the number of fuzzy rules. Too many fuzzy rules will increase the complexity of the model, reduce interpretability, and easily lead to overfitting, while too few fuzzy rules will affect the performance of the model.

The adaptive network-based fuzzy inference system (ANFIS) [11] is a well-known model of FNN based on the TSK fuzzy inference system [2]. The traditional ANFIS uses the grid-based method for structure identification, that is, for features of size M, the number of fuzzy rules is M^H when the fixed H-grid method is used. In terms of parameter identification, the gradient descent method, the least square method, or a combination of both is adopted by the ANFIS. For ANFIS and its variant models [11–13], the number of fuzzy rules is usually fixed so that the number of rules can be very large for data with high characteristic dimensions.

To alleviate "rule explosion", some examples from the literature [14,15] transfer input variables to a new feature space in advance by feature dimension reduction, such as the principal component analysis (PCA), and then carry out the structure and parameter identifications in this new space. For example, [15] restricts the maximum feature dimension to five and utilizes the PCA if the feature number exceeds five. Some studies [16–20] use methods based on clustering, such as fuzzy c-means (FCM) [21], Gustafson–Kessel clustering [22], and k-nearest neighbor clustering (KNN), to obtain a small number of fuzzy rules and then use local search methods, such as the gradient descent method, linear least square method (LLS), Levenberg–Marquardt (LM) method, extreme learning machine (ELM), and so on, to adjust the parameters. For example, [17] proposes a fuzzification method for the FNN based on Bayesian clustering. In that paper, a fuzzification technique based on the concepts of FCM is used, but with a Bayesian approach to optimize the assignment processing. For methods based on clustering, the clustering number often needs to be determined in advance. Considering that the clustering number has a great impact on the performance and interpretability of the model, some studies [23–25] utilize cross-validation or clustering validity indicators to determine it, but this is limited in effectiveness [26]. How to interpret the obtained clustering is also an essential problem. It has been pointed out that the fuzzy partitions obtained through clustering may overlap and lack semantics [27]. Some studies [28,29] remove redundant and unnecessary rules through rule reduction and rule pruning to reduce the number of rules to a certain extent. For example, a growing-and-pruning algorithm (GP) is proposed in the literature [29]. In GP, new rules are added, and useless rules are eliminated through a sensitivity analysis of the model output. Some works [30,31] alleviate the problem of rule explosion by using a hierarchical structure. For example, [30] proposes a novel hierarchical hybrid FNN to represent systems with mixed-input variables. Several fuzzy sub-systems on the lower level randomly aggregate several discrete input variables into intermediate outputs, and a neural network whose input variables consist of continuous input variables and intermediate variables is the higher layer, thereby reducing the input dimension and the number of fuzzy rules. However, there is no general approach to selecting suitable discrete features for combination.

In the studies of FNNs, structure and parameter identifications are usually carried out separately, that is, the numbers of fuzzy rules and fuzzy partitions are determined first, and then the parameters of the rules are adjusted. Parameter identification is generally divided into two stages: the adjustment of antecedent parameters and the adjustment of consequent parameters, and different methods are selected according to the characteristics of these two parameters. This learning strategy has the advantage of low time complexity, but it cannot capture the internal correlation of various parameters and so it is difficult to find the optimal solution in the whole parameter space. To coordinate structure identification and parameter identification, meta-heuristic search methods such as particle swarm optimization algorithms and evolutionary algorithms are considered in some of the literature [32–34]. In these algorithms, all parameters to be learned, such as the number of rules and parameters of membership functions (MFs), are simultaneously encoded into a long and complex chromosome for joint optimization. For example, a self-organizing FNN based on the genetic algorithm is proposed in the literature [33], and a hybrid algorithm based on genetic algorithms, backpropagation, and recursive least square estimation is adopted to adjust all parameters, including the number of fuzzy rules. Moreover, the multi-objective evolutionary algorithm is regarded as a cooperative method for structure identification and parameter identification, and it has been used to construct FNNs with high prediction accuracy and a simple structure [35,36]. However, these meta-heuristic search methods have high requirements for memory and computing resources.

Based on the previous study [37], a fuzzy neural network regression method (FNNR) with high precision and a compact structure is proposed. Compared with traditional FNNs, the proposed FNNR changes the structure and training mode of the network so that the number of fuzzy rules and the number of fuzzy partitions can be limited effectively by the gradient descent method, thereby alleviating the problem of rule explosion. In the FNNR, the optimization of all parameters, including the fuzzy partition number, the fuzzy rule number, parameters of antecedents, and consequent parameters, is incorporated into the training process, and these parameters are adjusted synergistically by the gradient descent method. On this basis, the alternate training strategy is designed and utilized for different types of parameters to help the algorithm converge to the global optimal solution. To further enhance the interpretability, regularization terms are designed from fuzzy rule level and fuzzy partition level to guide the model to realize high precision with a simple structure and clear semantics. By comparisons with some representative regression models based on fuzzy rules and the classical regression method, it is proven that the proposed FNNR can achieve high prediction accuracy with high interpretability.

The main contributions are as follows:

1. A new structure and training mode of the FNN is designed for regression problems, and thus the numbers of fuzzy rules and fuzzy partitions can be learned automatically by the gradient descent method, thereby eliminating the implicit relationship between the number of rules and the number of features and fuzzy partitions, meanwhile effectively alleviating the problem of rule explosion.
2. The structure and parameter identifications of the FNN are considered as a whole, which means that the number of fuzzy partitions, the number of fuzzy rules, the MF parameters of antecedents, and the parameters of the consequent are adjusted and tuned at the same time. On this basis, an alternate training strategy is designed to help with the algorithm convergence and find the optimal solution in the whole parameter space without any pre-processing or post-processing.
3. The interpretability of the model is measured from both the fuzzy rule level and fuzzy partition level, and the measurement is introduced to the training process in terms of regularization. Therefore, the trained model has high precision with a simple structure and clear semantics.

The remainder of this paper is organized as follows. The TSK fuzzy rules and the definition of the fuzzy neural network classifier (FNNC) that we studied earlier are introduced in Section 2. In Section 3, the methodology of the FNNR is proposed. Section 4 discusses the experimental results of the comparisons of the proposed FNNR with other benchmark methods. Conclusions and future works are offered in Section 5.

2. Preliminaries

2.1. The TSK Fuzzy Rules

The Takagi–Sugeno–Kang (TSK) fuzzy system proposed by Takagi, Sugeno, and Kang [2] is one of the most famous fuzzy systems with a simple structure and a good nonlinear approximation ability. For a TSK fuzzy system with multiple inputs and a single output, the form of fuzzy rules is shown in Equation (1):

$$\text{Rule } R^k: \text{if } x_1 \text{ is } A_1^k \text{ and} \ldots \text{and } x_M \text{ is } A_M^k, \text{then } \bar{y}^k = p_0^k + p_1^k x_1 + \ldots + p_M^k x_M \quad (1)$$

where R^k denotes the k^{th} rule, and $k = 1,\ldots,K$. x_m represents the m^{th} input fuzzy variable (feature), and $m = 1,\ldots,M$. $A_m^k = \{< x, \mu_{A_m^k}(x) > | x \in U\}$ is the fuzzy set of x_m, and $U \in [0,1]$. $\mu_{A_m^k}:[0,1] \to [0,1]$ denotes the MF of A_m^k, $\bar{y}^k$ is the consequent output of R^k, and p_m^k represents the linear function coefficients of the consequent. The "and" is the connective of the rule. The antecedents of the fuzzy rule can also be connected by the connective "or", which can be realized by simply replacing the "and" with the "or" in Equation (1).

For the sample data $\boldsymbol{d} = (\boldsymbol{x}, y)$, the fuzzy reasoning method of the TSK fuzzy system is shown below:

(1) Calculate the firing strength. The firing strength f^k of the feature vector $\boldsymbol{x}$ on R^k shown in Equation (1) is calculated as follows:

$$f^k = \mu_{A_1^k}(x_1) \wedge \ldots \wedge \mu_{A_M^k}(x_M) \tag{2}$$

where $\wedge$ represents the fuzzy intersection operator (*T*-norm operator). If the connective is "or", you just need to change the fuzzy intersection operator of Equation (2) to the fuzzy union operator (*S*-norm operator), which is expressed by $\vee$.

(2) Calculate the normalized firing strength. The normalized firing strength $\dot{f}^k$ of the feature vector $\boldsymbol{x}$ on R^k is calculated as follows:

$$\dot{f}^k = f^k \Big/ \sum_{k=1}^{K} f^k \tag{3}$$

(3) Calculate the output. The output $\overline{y}$ of the feature vector $\boldsymbol{x}$ on *K* fuzzy rules is calculated as follows:

$$\overline{y} = \sum_{k=1}^{K} \dot{f}^k \overline{y}^k \tag{4}$$

2.2. The FNNC

The FNNC consists of three different layers: the fuzzification layer, the fuzzy logic layer, and the classification layer.

- The fuzzification layer is utilized to translate crisp input features into fuzzy variables. Gaussian MFs are utilized. The parameters of Gaussian MFs are determined based on experience before the training and remain unchanged during the training.
- The fuzzy logic layers are used to represent fuzzy rules. Let $r_{ij} \in \{0,1\}$ denote the parameter of fuzzy logic layers, where *j* is the *j*th node of the fuzzy logic layer, and *i* is the *i*th node of the previous layer (the same below). The nodes of fuzzy logic layers represent "and" and "or" connectives in fuzzy rules through fuzzy intersection operators and fuzzy union operators. Through the connecting and stacking of multiple fuzzy logic layers, complex fuzzy rules can be represented.
- The classification layer is for integrating the outputs of fuzzy logic layers and giving the final classification. The number of nodes in the classification layer is the same as the number of class labels.

To perform structure and parameter identifications through the iterative algorithm of the neural network, the FNNC-d, the symmetrical structure of the FNNC is designed. The difference between the two lies in the parameters of fuzzy logic layers. In the FNNC-d, the parameters of fuzzy logic layers denoted by $\hat{r}_{ij}$ are continuous real numbers, i.e., $\hat{r}_{ij} \in [0,1]$. The two parameters can be converted by the function $q:[0,1] \to \{0,1\}$:

$$q(\hat{r}_{ij}) = r_{ij} = \begin{cases} 0 & \hat{r}_{ij} \le 0.5 \\ 1 & \hat{r}_{ij} > 0.5 \end{cases} \tag{5}$$

On this basis, a parameter conversion method during the training process is designed. The FNNC can be utilized for training, testing, and interpreting, while the FNNC-d is only used for training. The gradient-updating formula of the FNNC is shown in Equation (6):

$$\theta^{t+1} = \theta^t - \eta \frac{\partial \mathcal{L}(\overline{Y})}{\partial \overline{Y}} \cdot \frac{\partial (\hat{Y})}{\partial \theta^t} \tag{6}$$

where θ^t represents the parameter of the FNNC-d at step *t*, η is the learning rate, and $\mathcal{L}(\cdot)$ is the loss function. $\overline{Y}$ and $\hat{Y}$ refer to the outputs of the FNNC and the FNNC-d, respectively.

3. The FNNR

In this chapter, a novel regression method of FNN named FNNR is proposed. In the FNNR, the structure identification and parameter identification are completed cooperatively, which gives the model high prediction accuracy with a simple structure and clear semantics.

3.1. The Structure of the FNNR

The structure of the FNNR is shown in Figure 1. The model consists of five different layers: the fuzzification layer, fuzzy logic layers, normalization layers, consequent layers, and the sum layer. Each layer contains several neurons, and the neurons are connected by edges.

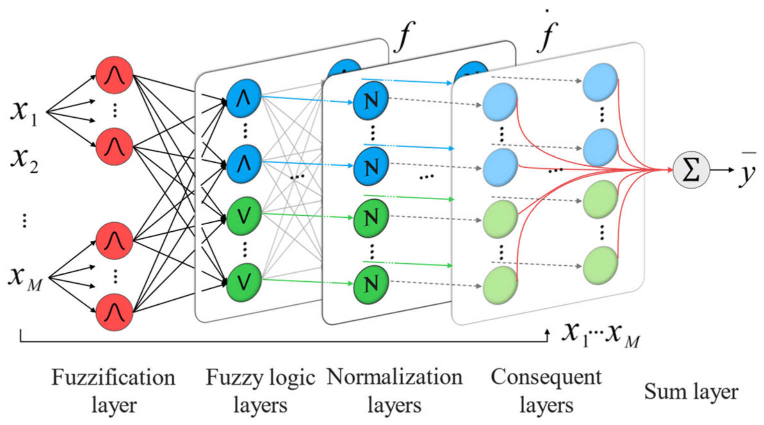

Figure 1. The structure of the FNNR.

Let L denote the layer number of the FNNR, where $L = 1 + G + G + G + 1$. G is the layer number of fuzzy logic layers and $G \geq 1$. The first layer is the fuzzification layer for translating crisp input features into fuzzy variables. The middle *G* layers are fuzzy logic layers (same as the fuzzy logic layers in the FNNC). The nodes "$\wedge$" and "$\vee$" refer to the fuzzy intersection operator and the fuzzy union operator, respectively, and the output of each node is the firing strength f^k of the corresponding rule. The first fuzzy logic layer accepts the outputs of the fuzzification layer as the inputs, and the multiple fuzzy logic layers represent complex fuzzy rules by connecting with each other. The number of nodes "$\wedge$" and "$\vee$" in each fuzzy logic layer and the layer number (*G*) can be determined according to the complexity of the task. Skip connections are added between fuzzy logic layers to conveniently express concise rules. The subsequent *G* layers are normalization layers for normalizing the firing strength f^k output by the node in fuzzy logic layers. The next *G* layers are consequent layers for representing and calculating the consequent outputs of TSK rules. The last layer is the sum layer, which is used to calculate the final prediction. The normalization layers, consequent layers, and the sum layer are collectively called output layers.

The FNNR is a novel FNN model. In terms of form, like the neural network, the FNNR is composed of multi-layer neurons. The knowledge is acquired from the sample data and parameters are adjusted through training. In terms of the calculation process, the FNNR equals a TSK fuzzy system that can learn automatically. By designing the functions of different nodes, the superpositions of neurons between layers are transformed into the fuzzy logic operation, the fuzzy rule combination, and the fuzzy reasoning. Therefore, the training process of the FNNR is the process of structure and parameter identifications of the TSK fuzzy system. The key details of the FNNR are introduced in detail below.

3.1.1. The Fuzzification Layer

The fuzzification layer is utilized to translate the crisp input feature vector $\boldsymbol{x} = (x_1, ..., x_M)$ into fuzzy linguistic variable values (fuzzy sets). Let H denote the number of MFs of each feature, then there are $M \times H$ nodes in the fuzzification layer. The node of the fuzzification layer is represented by A_{mh}, which refers to the h^{th} fuzzy set of x_m, $m = 1, ..., M$, and $h = 1, ..., H$. The Gaussian MFs are used to represent the fuzzy sets. To improve the prediction accuracy, the parameters of Gaussian MFs are adjusted during the training. In addition, considering that different features may use different fuzzy partitions, an additional fuzzy partition parameter, $e_{mh} \in \{0,1\}$, is introduced, which refers to whether the h^{th} fuzzy set of x_m is retained: $e_{mh} = 1$ means keeping the h^{th} fuzzy set, otherwise, the h^{th} fuzzy set is discarded. Therefore, the output of the node A_{mh} is shown in Equation (7):

$$\mu_{A_{mh}}(x_m) = e_{mh} \exp(-\frac{(x_m - c_{mh})^2}{2\sigma_{mh}^2}) \tag{7}$$

where $\mu_{A_{mh}}(x_m) \in [0,1]$ refers to the fuzzy value of x_m on the fuzzy set A_{mh}, and c_{mh} and σ_{mh} are the mean and standard deviation of the Gaussian MF, respectively. The parameters of the fuzzification layer are shown in Figure 2.

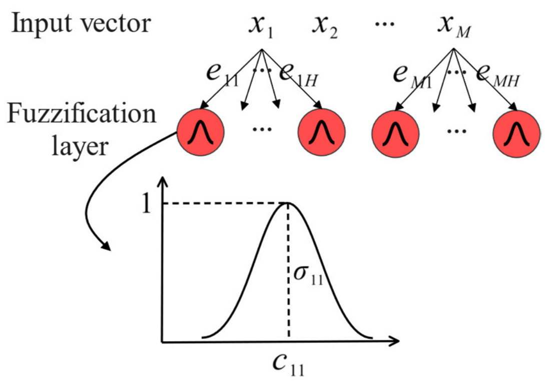

Figure 2. The parameters of the fuzzification layer of the FNNR.

It should be noted that the introduction of the fuzzy partition parameter e_{mh} can not only control the number of fuzzy sets used for each feature but can also indirectly control the number of input variables. Specifically, when the fuzzy partition parameters corresponding to x_m are all zeros, that is, $e_{m1} = 0, ..., e_{mH} = 0$, then x_m is equivalent to being discarded and has no contribution to the prediction result.

To enable e_{mh}, a discrete parameter, to be trained by the gradient descent and the back propagation algorithm, the FNNR-d is designed as the symmetrical model of the FNNR, and the fuzzy partition parameter of the FNNR-d, $\hat{e}_{mh}$, is set to the continuous value of the real-value range of [0, 1]. The same as Equation (5), $\hat{e}_{mh}$ and e_{mh} can be transformed by the function $q(\cdot)$:

$$q(\hat{e}_{mh}) = e_{mh} = \begin{cases} 0 & \hat{e}_{mh} \le 0.5 \\ 1 & \hat{e}_{mh} > 0.5 \end{cases} \tag{8}$$

The FNNR and the FNNR-d share the parameters of Gaussian MFs in the fuzzification layer.

3.1.2. The Output Layers

The output layers consist of normalization layers, consequent layers, and one sum layer. They are used to integrate the outputs of nodes in fuzzy logic layers and give the final prediction result. Let *K* donate the number of nodes in the fuzzy logic layers.

The normalization layer is used to normalize the firing strengths. As shown in Figure 1, normalization layers accept the firing strength f^k output by the node in fuzzy logic layers and get the normalized firing strength $\dot{f}^k$, which corresponds to Equation (3), and $k = 1,\ldots,K$. From Figure 1, the number of nodes in normalization layers is the same as that in the fuzzy logic layers, and there is a corresponding relationship between the normalization node and the fuzzy logic node.

The consequent layer is for computing the consequent output $\overline{y}^k$ of each rule. As can be seen from Figure 1, the layer number of consequent layers is the same as that of the fuzzy logic layers, and their nodes correspond one by one. Each node of consequent layers accepts the feature vector $x_1,\ldots,x_M$ as the input (see the arrow at the bottom of Figure 1) and calculates the consequent output $\overline{y}^k$ of the corresponding rule according to Equation (9):

$$\overline{y}^k = p_0^k + p_1^k x_1 + \ldots + p_M^k x_M \tag{9}$$

Therefore, the parameters of consequent layers are the linear function coefficients of TSK rules, which are denoted by W_P.

The sum layer is used to integrate the normalized firing strength $\dot{f}^k$ and the consequent output $\overline{y}^k$ of each rule to obtain the final prediction. The normalization layers pass $\dot{f}^k$ to consequent layers (see the gray arrow in Figure 1). Then, consequent layers take $\dot{f}^k$ together with the output $\overline{y}^k$ as the inputs of the sum layer and, finally, the summation operation is complete in the sum layer, which corresponds to Equation (4). It should be noted that the normalization layers and the sum layer have no trainable parameters, and they only complete the normalization operation and the summation operation, respectively. Therefore, the parameter scale of output layers is $M \cdot K$, where *M* is the number of input features, and *K* is the number of nodes in fuzzy logical layers.

Since the trainable parameters (W_P) of output layers are values of the continuous interval, the FNNR and the FNNR-d share the output layers.

3.2. Training

3.2.1. The Design of the Loss Function

Similarly to the FNNC, the FNNR needs to find the gradient direction with the help of its symmetrical model, FNNR-d, thus the updating of parameters is similar to Equation (6), except that the loss function is different. For the FNNR, since the structure identification and parameter identification are coordinated, there are altogether four kinds of parameters that need to be trained through gradient descent algorithm, which are the parameters of Gaussian MFs, $W_G(\boldsymbol{c},\boldsymbol{\sigma})$, the fuzzy partition parameters, $W_E(\boldsymbol{e})$, the parameters of fuzzy logic layers, $W_R(\boldsymbol{r})$, and the consequent parameters, $W_P(\boldsymbol{p})$. The change in these parameters will not only influence the prediction accuracy of the model but also greatly affect its interpretability. Therefore, the loss function is divided into two parts: $\mathcal{L}_{acc}(\cdot)$ and $\mathcal{L}_{inter}(\cdot)$, which are utilized to calculate the loss of prediction accuracy and interpretability, respectively.

The loss function of the prediction accuracy $\mathcal{L}_{acc}(\cdot)$ is calculated as shown in Equation (10):

$$\mathcal{L}_{acc}(\overline{Y}) = \mathrm{MSE}(Y, \overline{Y}(X,W)) \tag{10}$$

where the mean square error (MSE) function is used. $W = (W_G, W_P, W_E, W_R)$ represents the parameters of the FNNR, including parameters of Gaussian MFs (W_G), consequent parameters (W_P), discrete parameters of fuzzy partitions (W_E), and discrete parameters of fuzzy logic layers (W_R). $\overline{Y}$ is the output of the FNNR, which is the final prediction, and Y is the label.

The measure of the model's interpretability is considered from two levels: the interpretability at the fuzzy rule level and the interpretability at the fuzzy partition level as shown in Table 1. Among them, the fewer rules extracted from the model and the fewer antecedents of the rule, the more concise and explainable fuzzy rules are. The fewer input variables and MFs the model uses and the more complementary the fuzzy partition is, the clearer the fuzzy partition is and the more interpretable the model is. Here, the complementarity [38] refers to the fact that the sum of the fuzzy values of input features on all fuzzy partitions is close to one.

Table 1. Interpretability measures of the FNNR.

Fuzzy Rule Level	**Fuzzy Partition Level**
Number of Rules	Number of MFs
Number of Antecedents	Number of Input Features
	Complementarity

Therefore, the loss function $\mathcal{L}_{inter}(\cdot)$ that measures the interpretability of the FNNR is revealed in Equation (11):

$$\mathcal{L}_{inter}(\hat{W}) = \lambda_1 \varphi_1(W_G) + \lambda_2 \|\hat{W}_E\|_2^2 + \lambda_3 \|\hat{W}_R\|_2^2 \tag{11}$$

where $\hat{W} = (W_G, W_P, \hat{W}_E, \hat{W}_R)$ are parameters of the symmetrical model FNNR-d. λ_1, λ_2, and λ_3 are the regularization coefficients for parameters of Gaussian MFs (W_G), parameters of fuzzy partitions ($\hat{W}_E$), and parameters of fuzzy logic layers ($\hat{W}_R$), respectively. $\varphi_1(\cdot)$ is the function that measures the complementarity of fuzzy sets as shown in Equation (12):

$$\varphi_1(W_G) = \sum_x \left(\sum_{h=1}^{H} \mu_{A_h}(x) - 1\right)^2 \tag{12}$$

where x is the value of the input feature, A_h refers to the hth fuzzy set of x, and H is the fuzzy set number of x.

It can be observed from Equation (11) that the first item of $\mathcal{L}_{inter}(\cdot)$ can help to enhance the complementarity of fuzzy partitions during the training, the second item can help the model to reduce the number of input variables and MFs in the training process, and the third item can help to reduce the number of fuzzy rules and antecedents in the training process.

In conclusion, the formula of parameter updating is shown in Equation (13):

$$\hat{w}^{t+1} = \hat{w}^t - \eta\left(\frac{\partial \mathcal{L}_{acc}(\overline{Y})}{\partial \overline{Y}} \cdot \frac{\partial(\hat{Y})}{\partial \hat{w}^t} + \frac{\partial \mathcal{L}_{inter}(\hat{w}^t)}{\partial \hat{w}^t}\right) \tag{13}$$

where $\hat{w} \in \hat{W}$ represents the parameter of the symmetrical model FNNR-d. $\overline{Y}$ and $\hat{Y}$ are the outputs of the FNNR and the FNNR-d, respectively.

3.2.2. The Alternate Training Strategy

As mentioned above, the FNNR and its symmetrical model FNNR-d contain four types of trainable parameters, where parameters of Gaussian MFs W_G and consequent parameters W_P are shared, while parameters of fuzzy partitions and fuzzy logic layers are different, which can interchangeable by $q(\cdot)$ (see Equations (5) and (8)). According to $q(\cdot)$, when the values of fuzzy partition parameters $\hat{W}_E$ and fuzzy logic layer parameters $\hat{W}_R$ in the FNNR-d cross 0.5, the corresponding discrete fuzzy partition parameters W_E and fuzzy logic layer parameters W_R in the FNNR will jump from 0 to 1 (or from 1 to 0), and then the parameters of Gaussian MFs W_G and the consequent parameters W_P will change dramatically. Therefore, when the above four kinds of parameters are trained together, we find that the model is difficult to converge during the training process: $\hat{W}_E$ and $\hat{W}_R$ oscillate around 0.5, which leads to the constant oscillation of W_G and W_P, making it difficult to find the optimal solution. To solve this problem, an alternate training strategy is designed and adopted in training as shown in Algorithm 1.

Algorithm 1: Alternate Training Strategy

```
input: The dataset, $D$; the number of epoches for joint training, $E_1$; the number
of epoches for fixing fuzzy logic layers, $E_2$; the number of epoches for fixing fuzzy
partitifon parameters, $E_3$.
output: A trained FNNR model, FNNR
begin
    initialize the parameters of model
    for $i < v$ do
        training and updating all the parameters with $D$ for $E_1$ epoches\;
        training and updating all the parameters except for $\hat{W}_R$ with $D$ for
        $E_2$ epochs;
        training and updating all the parameters except for $\hat{W}_E$ with $D$ for
        $E_3$ epochs;
        $i = i + 1$;
    end
    return FNNR
end
```

It can be observed from Algorithm 1 that the whole training cycle is divided into v rounds and each round is divided into three stages: the stage of joint training, the stage of fixed fuzzy logic layers, and the stage of fixed fuzzy partition parameters. Utilizing three stages of alternate training, the problem of the model brings difficult to converge, caused by the oscillation of fuzzy partition parameters and fuzzy logic layer parameters, can be alleviated.

4. Experiments

To verify the regression performance of the proposed method, the FNNR is compared with the representative regression methods, based on fuzzy rules proposed recently, and the classical regression algorithm on benchmark datasets.

4.1. Experimental Design

We have the following questions in mind while designing and conducting the experiments:

(1) How does the FNNR perform when compared with other state-of-the-art regression methods?
(2) What roles do the training modes of the fuzzification layer, the design of the output layers, and the strategy of alternate training play in the FNNR?
(3) How much do different regularization coefficients affect the final prediction of the FNNR?
(4) How explainable is the FNNR?

To answer (1), we compare the performance of the FNNR with several benchmark methods. To answer (2), we conduct some ablation studies on the proposed model. To answer (3), we change the regularization coefficients to make comparisons. To answer (4), the fuzzy rules used by the FNNR to make predictions are visually displayed on two datasets.

4.1.1. Datasets

A total of 28 real regression datasets are selected from KEEL [39] repository as baseline datasets. These datasets have different feature numbers (from two to forty) and sample numbers (from 337 to 40,768). According to the dimension of features, these 28 datasets are divided into two categories: 11 low-dimensional datasets and 17 high-dimensional datasets. Supplementary S1 shows detailed information on the two types of datasets. For the low-dimensional datasets, the number of features is small (all less than seven), and the number of samples is also relatively small, which is up to 4052, so the regression tasks on low-dimensional datasets are relatively simple. For high-dimensional datasets, the number of features is large, especially for the last four datasets, and the feature numbers are all over 20. Considering that fuzzy rules are better at fitting data with small feature dimensions [14], the regression performance of the model is challenged. In addition, in high-dimensional datasets, some of them have small sample sizes, such as FOR and BAS, which increases the risk of overfitting. Finally, there are quite a few datasets with large sample sizes, such as CAL, MV, and HOU, which may consume a lot of computing resources.

4.1.2. Parameter Settings for the FNNR

In the FNNR, Gaussian MFs used in the fuzzification layer are set as the uniform MFs in the domain [0, 1], that is, the mean vector of Gaussian functions is shown in Equation (14):

$$\boldsymbol{c}=\begin{cases}(0,1) & H=2\\ (0,1/(H-2),1) & H=3\\ (0,1/(H-2),2/(H-3),\ldots,1) & H>3\end{cases} \tag{14}$$

Each MF uses the same standard deviation: $\sigma=2\cdot(1/H)^2$. Fuzzy partition parameters are all initialized to 1 s. To enhance the model interpretability as much as possible, the initial number of fuzzy sets for each feature, H, is set to seven. Considering that the prediction task on low-dimensional datasets is relatively simple, parameters of MFs and fuzzy partitions in the FNNR on low-dimensional datasets are fixed and unchanged with $H \in \{3,5\}$. The number of fuzzy logic layers is chosen from $\{1,2\}$. Depending on the regression difficulties of different datasets, the number of nodes in each fuzzy logic layer ranges from two to fifty. We utilize the Adam method and the MSE loss function for the training process. The round number v for the alternate training strategy is set to three, and the cycle numbers of three training stages are set to 100 in each round, i.e., $E_1,E_2,E_3=100$. The ranges of regularization coefficients in Equation (11) are as follows: $\lambda_1\in\{0,\text{1e-4},\text{1e-2},1\}$, $\lambda_2\in\{\text{1e-3},\text{1e-6},1\}$, and $\lambda_3\in\{\text{1e-2},\text{1e-4},\text{1e-6},\text{1e-8},\text{1e-10}\}$. The min-max normalization is carried out on features and labels. For a fair comparison, the prediction results of the proposed method are inversely normalized and the MSEs with the original labels are calculated and recorded.

4.1.3. Experimental Settings

- Benchmark Methods

To evaluate the regression performance of the proposed FNNR, it is compared with some representative regression methods based on fuzzy rules and one classical regression algorithm, including the following seven methods:

a. The decision tree (DT) [40];
b. The training algorithm for the TSK fuzzy system based on mini-batch gradient descent with regularization, droprule, and adabound (MBGD-RDA) [14];
c. The learning algorithm of TSK fuzzy rules based on evolutionary learning (FRULER) [41];
d. The learning algorithm of the TSK fuzzy system based on multi-objective evolutionary algorithms (METSK-HD) [35];

e. The learning algorithm of the zero-order TSK fuzzy system based on Apriori and local search methods (Freq-SD-LSLS) [42];
f. The learning algorithm of Mamdani fuzzy rules based on multi-objective evolutionary learning algorithms (MOKBL + MOMs) [36];
g. Disjunctive fuzzy neural network (DJFNN), a new splitting-based approach to designing a TS fuzzy model [10].

The experimental results of the benchmark methods are directly cited from [10].

- Evaluation Metrics

In this paper, the MSE on the test dataset is adopted to evaluate the regression performances of different methods, which is shown in Equation (15):

$$MSE = \frac{1}{2N_{test}} \sum_{n=1}^{N_{test}} (\overline{y}_n - y_n)^2 \tag{15}$$

where N_{test} is the sample size of the test data. $\overline{y}_n$ and y_n refer to the prediction and the label of the n^{th} sample, respectively. In this paper, all the experiments are repeated 10 times on each dataset, and the average MSEs are reported.

- The Significance Testing

To further explore whether the observed differences are statistically significant, the Friedman test [43] for multiple comparisons and the Bonferroni–Dunn post hoc test [44] to identify pairwise differences are applied.

4.2. Result Analysis

The average ranks (AvgR) and average MSEs of the proposed FNNR and the other four regression methods on low-dimensional test data are reported in Table 2, where the results of ELE1, DELAIL, and DELELV should be multiplied by 10^5, 10^{-8}, and 10^{-6} (the same below). The minimum MSE on each dataset is highlighted in bold.

Table 2. The average test MSEs of FNNR, DT, MBGD-RDA, FRULER, and DJFNN on low-dimensional datasets.

Datasets	DT	MBGD-RDA	FRULER	DJFNN	FNNR (Ours)
ELE1	2.495	2.082	2.012	2.242	**1.504**
PLA	3.708	1.176	1.219	1.116	**1.095**
QUA	**0.0178**	0.0180	0.0181	0.0179	0.0180
ELE2	1.1×10^5	13,677	6729	4107	**3922**
FRIE	5.486	3.654	0.731	0.788	**0.605**
MPG6	6.258	4.416	3.727	4.083	**3.696**
DELAIL	1.735	1.502	1.458	**1.362**	1.456
DEE	0.119	0.085	**0.080**	0.082	**0.080**
DELELV	1.216	1.059	1.045	**1.008**	1.015
ANA	**0.003**	0.086	0.008	0.004	0.004
MPG8	6.652	4.325	4.084	4.315	**3.095**
AvgR	4.273	3.909	2.727	2.364	**1.455**

We can conclude from Table 2 that:

1. The proposed FNNR achieves the minimum average MSEs on seven of eleven datasets among the models involved. On three of the remaining four datasets, FNNR still exhibits the second-best performance. It proves that the proposed FNNR method has a significant performance advantage on low-dimensional and simple tasks. Considering that parameters of MFs and fuzzy partitions in the fuzzification layer are not adjusted during the training process on low-dimensional datasets, there is still a lot of room for performance improvement of the FNNR.

2. The FNNR shows great improvement over the other four approaches in terms of regression performance. On some datasets, such as ELE1, FRIE, and MPG8, the FNNR reduces the MSEs by over 20% compared with the recently proposed DJFNN. Moreover, the average MSEs are more than 4% lower than the DJFNN on datasets ELE2 and MPG6.
3. The significance tests are conducted on the results shown in Table 2. The Friedman test suggests rejecting the H_0 hypothesis ($F_F = 5.825 > 2.091$) for a significance level of 0.1 with (4,40) degrees of freedom. This suggests that on low-dimensional datasets, there are significant differences between at least two methods across the benchmark. The Bonferroni–Dunn post hoc test suggests that the regression performance of the FNNR is significantly different from that of DT, MBGD-RDA, and FRULER, while the performance of the FNNR and the DJFNN are equivalent.

The average ranks and average test MSEs of the proposed FNNR and all of the seven regression methods on high-dimensional datasets are reported in Table 3, where the results of CAL, BAS, HOU, ELV, PUM, and AIL should be multiplied by 10^9, 10^5, 10^8, 10^{-6}, 10^{-4}, and 10^{-8} (the same below). The minimum MSE on each dataset is highlighted in bold.

Table 3. The average test MSEs of FNNR, DT, MBGD-RDA, FRULER, DJFNN, METSK-HD, Freq-SD-LSLS, and MOKBL + MOMs on high-dimensional datasets.

Datasets	**DT**	**MBGD-RDA**	**FRULER**	**DJFNN**	**METSK-HD**	**Freq-SD-LSLS**	**MOKBL +MOMs**	**FNNR (Ours)**
ABA	2.957	2.518	2.393	2.253	2.392	2.476	2.401	**2.035**
CAL	3.303	2.449	2.110	2.050	1.710	2.385	2.660	**1.674**
CON	51.27	55.13	20.60	18.26	23.89	17.04	27.42	**13.27**
STP	1.764	2.733	0.353	0.392	0.387	0.725	0.660	**0.199**
WAN	6.835	1.258	0.888	**0.728**	1.189	1.025	1.600	0.741
WIZ	4.713	0.814	0.663	0.755	0.944	0.955	1.580	**0.641**
MV	4.071	10.18	0.083	0.006	0.061	0.273	0.093	**0.003**
FOR	3209	2009	2214	2908	5587	2317	**2006**	2249
MOR	0.160	0.013	0.007	0.003	0.013	0.026	0.015	**0.002**
TRE	0.167	0.032	0.027	**0.023**	0.038	0.054	0.041	**0.023**
BAS	2.876	2.638	3.0e5	3.309	3.688	3.120	**2.570**	2.627
HOU	9.121	11.49	8.005	6.755	8.640	6.847	9.110	**6.535**
ELV	11.52	34.45	2.934	**2.360**	7.020	10.00	10.70	2.399
CA	9.972	48.85	4.634	2.815	4.949	61.97	4.670	**2.089**
POLE	149.6	471.8	110.9	119.7	61.02	541.8	93.96	**19.79**
PUM	1.501	3.636	0.367	0.223	0.287	1.520	0.270	**0.198**
AIL	2.976	6.9e8	1.404	1.309	1.510	4.581	1.821	**1.302**
AvgR	6.941	6.294	3.647	2.765	4.353	5.706	4.824	**1.353**

We can conclude from Table 3 that:

1. The FNNR also shows great improvement over other approaches in terms of regression performance on high-dimensional tasks. On some datasets, such as STP, MV, and POLE, the FNNR reduces the errors by about 50% compared with the DJFNN. Moreover, the average MSEs are more than 20% lower than the DJFNN on datasets CON, MOR, and CA.
2. The proposed FNNR obtained the minimum MSEs on the last four datasets with high feature dimensions, which indicates that the selection of features and antecedents is completed flexibly through trainable parameters of fuzzy partitions and fuzzy logic layers in the FNNR. For datasets BAS and FOR, which are easy to overfit, although the FNNR does not get the optimal performance among all the methods, there is no significant difference between the average error of the FNNR and the best one, with increases of 2% and 10.8%, respectively. It indicates that the proposed method can

avoid overfitting to a certain extent. On datasets with large sample sizes like MV, CAL, and HOU, the FNNR achieves the minimum MSEs, showing that the FNNR also has good performance in handling regression tasks with large samples.

The Friedman test suggests rejecting the H_0 hypothesis ($F_F = 20.024 > 1.771$) for a significance level of 0.1 with (7112) degrees of freedom. This suggests that there are significant differences between at least two methods across the benchmark. The Bonferroni–Dunn post hoc test suggests that the regression performance of the FNNR is significantly different from that of the DJFNN ($1.412 > CD_\alpha = 1.188$).

4.3. Ablation Study

To illustrate the functions and effects of some key technologies in the FNNR, such as the training modes of the fuzzification layer, the design of the output layers, and the alternate training strategy, ablation studies are carried out. For fairness, the rest of the parts remain unchanged and the hyperparameters remain the same during ablation experiments.

4.3.1. The Ablation of Training Modes of the Fuzzification Layer

To illustrate the influence of training modes of parameters in the fuzzification layer on the regression performance, the following three different training modes are adopted:

a. The parameters of the fuzzification layer are fixed during the training.
b. Only the parameters of Gaussian MFs in the fuzzification layer are trained.
c. The parameters of Gaussian MFs and fuzzy partitions in the fuzzification layer are trained together.

On high-dimensional datasets, the average prediction MSEs of the FNNR using the above three training modes are shown in Figure 3. The FNNR achieves the minimum errors on 14 of 17 high-dimensional datasets using training mode c. Compared with training mode b, the regression performance of the FNNR with training mode c is slightly improved, for example, the MSEs on datasets ABA, CON, STP, FOR, and BAS are reduced by 4% to 22%. Compared with training mode a, the performance with training mode c is greatly improved, and the MSEs on datasets CON, STP, MV, FOR, MOR, CA, and POLE are reduced by 10% to 85%.

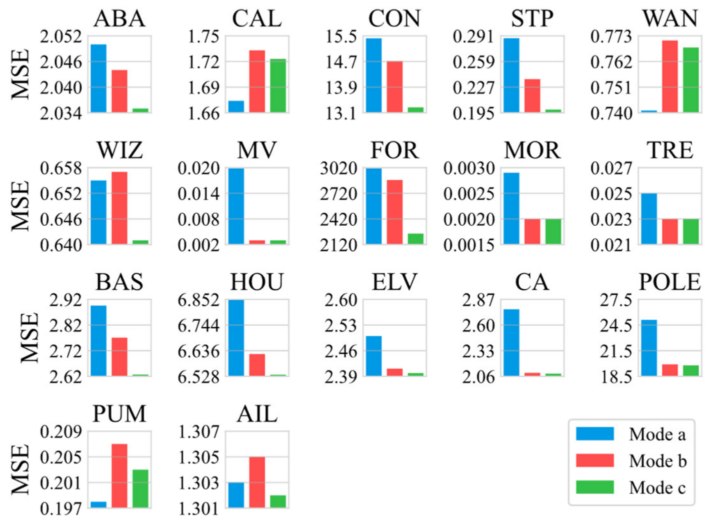

Figure 3. The average MSEs of the FNNR using different training methods on high-dimensional datasets.

Figure 4 reveals the fuzzy sets of each feature obtained by the above three training modes on the MOR dataset. Since the parameters of MFs are not modified and the same parameters of fuzzy partitions and MFs are used for all the features when using training mode a, the fuzzy sets of only one feature are displayed, and the fuzzy sets of the other features are the same.

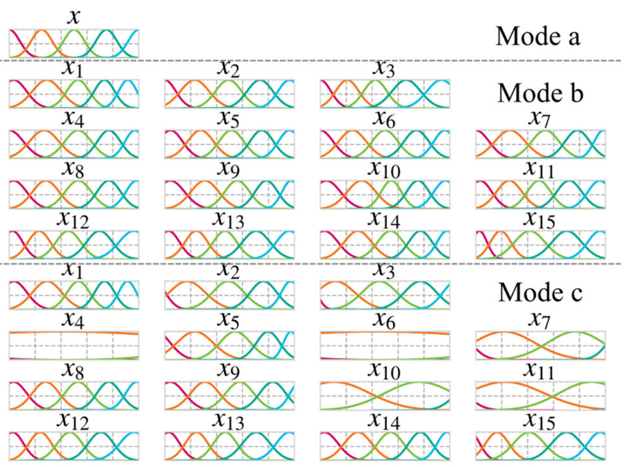

Figure 4. Fuzzy sets of each feature obtained by using three training modes on the MOR dataset.

As can be observed from Figures 3 and 4, training mode b enhances the fitting ability and improves the performance of the model by fine-tuning the parameters of MFs, and training mode c further reduces the test error of the model by discarding unimportant fuzzy partitions.

4.3.2. The Ablation of the Rule Type Represented by the Output Layers

The output layers of the FNNR consist of normalization layers, consequent layers, and the sum layer, which complete the representation of consequents in the TSK rules and the inference of prediction together. To verify the validity and rationality of the data fitting ability using TSK fuzzy rules, an ablation study is conducted on the rule type. In this study, the TSK fuzzy rules are replaced by Mamdani fuzzy rules [3] and the fuzzy rules for using nonlinear calculations in the consequents.

(1) The FNNR model with Mamdani fuzzy rules

Like TSK fuzzy rules, Mamdani rules are also a kind of fuzzy rule that are common and widely used. Different from the former, the antecedents and consequents of Mamdani rules are interpretable linguistic variables, so their interpretability is stronger. For convenience of representation, the original FNNR model is called FNNR-T, and the FNNR using Mamdani fuzzy rules is called FNNR-M.

The function of output layers in the FNNR-M is as the center average defuzzifier [45]. The output layers of the FNNR-M consist of one consequent layer and one sum layer, where the nodes of the consequent layer represent the fuzzy set of the output variable and the number of nodes is the fuzzy partitions number of the output variable, which is set to H_M. The parameters of the consequent layer are continuous real values in the interval [0, 1] that represents the rule weights. The consequent layer is fully connected with the fuzzy logic layers, and the output of each consequent node is the weighted sum of firing strengths of the rules whose consequent is the corresponding fuzzy set. For the FNNR-M, the parameter scale in the output layers is $H_M \cdot K$, where K is the number of nodes in fuzzy logic layers. Supplementary S2 shows more details on the FNNR-M.

(2) The FNNR model with rules whose consequents use nonlinear calculations

Considering that the fully connected network has high prediction accuracies in regression tasks, the consequent layers of the FNNR are replaced by fully connected layers, which is called the FNNR-F. In the FNNR-F, the output layers are composed of fully connected layers and one sum layer. The layer number of fully connected layers is set as $I_F(I_F \geq 1)$, and the number of nodes for each layer is set as n_F. The ReLU function is utilized as the activation function. Of course, for the FNNR-F, the prediction is the result of a complex weighted sum and nonlinear activation of the firing strengths, which reduces the model interpretability to a certain extent. The parameter scale of the output layers is $I_F \cdot (n_F + 1) + n_F \cdot K$. Supplementary S3 shows more details on the FNNR-F.

Table 4 shows the average MSEs of FNNR-M, FNNR-F, and FNNR-T on 28 datasets, where H_M is set as five, I_F is set as two, and n_F is set as twenty. The minimum error on each dataset is highlighted in bold.

Table 4. The average MSEs of EFNR-M, EFNR-F, and EFNR-T on 28 datasets.

Datasets	FNNR-M	FNNR-F	FNNR-T
ELE1	1.712	**1.361**	1.504
PLA	1.104	**1.062**	1.095
QUA	**0.0180**	0.0184	**0.0180**
ELE2	5882	**2585**	3922
FRIE	0.655	0.692	**0.605**
MPG6	**3.618**	3.669	3.696
DELAIL	1.513	**1.396**	1.456
DEE	0.077	**0.076**	0.080
DELELV	1.019	**0.928**	1.015
ANA	**0.003**	**0.003**	0.004
MPG8	3.402	**2.700**	3.095
ABA	2.033	**1.973**	2.035
CAL	1.750	**1.582**	1.674
CON	17.08	16.738	**13.27**
STP	0.299	0.321	**0.199**
WAN	**0.729**	0.834	0.741
WIZ	0.697	0.669	**0.641**
MV	0.031	0.221	**0.003**
FOR	3018	**2198**	2249
MOR	0.009	0.004	**0.002**
TRE	0.032	0.025	**0.023**
BAS	2.827	**2.521**	2.627
HOU	6.990	6.861	**6.535**
ELV	2.554	**1.987**	2.399
CA	3.769	3.491	**2.089**
POLE	36.16	**8.788**	19.79
PUM	0.326	**0.157**	0.198
AIL	1.354	1.317	**1.302**

As can be seen from Table 4, the regression performance of the FNNR-M is the worst: it can only achieve the minimum test errors on four of twenty-eight datasets. The FNNR-F has the best regression performance and can obtain the minimum test errors on 15 datasets. The performance of FNNR-T is middle-ranking with the minimum MSEs on 11 datasets. An analysis of the time complexity of the three models is given in Supplementary S4.

To improve the prediction accuracy of the FNNR-M and reduce the complexity of FNNR-F to enhance its interpretability, ablation analyses are performed on hyperparameters H_M and I_F, where $H_M \in \{3,5,7\}$, and $I_F \in \{1,2,3,4\}$. Table 5 illustrates the average MSEs of the above two models under each specified hyperparameter. The minimum error of each model on each dataset is highlighted in bold.

Table 5. The average prediction error of FNNR-M and FNNR-F on each dataset using different hyperparameters of H_M and I_F, respectively.

Datasets	FNNR-M						FNNR-F							
	3		5		7		1		2		3		4	
	Train	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
ELE1	2.833	**1.622**	1.917	1.712	2.514	1.690	1.767	1.527	1.098	**1.361**	1.517	1.528	1.375	1.397
PLA	1.116	**1.080**	1.113	1.104	1.105	1.087	1.106	1.076	1.089	**1.062**	1.086	1.077	1.103	1.069
QUA	0.016	0.019	0.017	**0.018**	0.017	0.019	0.017	0.018	0.017	**0.018**	0.017	0.019	0.017	0.019
ELE2	8523	8570	7207	**5882**	10,864	11,186	8603	9499	2600	**2585**	4478	4522	2904	3379
FRIE	0.600	**0.608**	0.629	0.655	0.601	0.621	0.596	0.725	0.609	0.692	0.606	**0.682**	0.606	0.704
MPG6	2.128	3.635	2.702	**3.618**	2.406	3.680	1.827	3.657	2.213	3.669	2.550	**3.623**	2.272	3.638
DELAIL	1.397	1.535	1.369	**1.513**	1.392	1.519	1.299	1.559	1.638	**1.396**	1.088	1.399	1.071	1.876
DEE	0.084	0.083	0.077	**0.077**	0.070	0.084	0.071	0.086	0.065	**0.077**	0.053	0.086	0.033	0.082
DELELV	1.018	**1.017**	1.008	1.019	1.008	1.021	0.991	1.006	0.883	**0.928**	0.954	0.997	0.948	0.995
ANA	0.003	**0.003**	0.003	**0.003**	0.003	0.004	0.003	**0.003**	0.002	**0.003**	0.002	**0.003**	0.002	**0.003**
MPG8	3.103	3.720	2.146	**3.402**	2.726	3.619	1.747	2.980	2.070	**2.700**	1.961	2.929	2.170	3.306
ABA	2.224	**2.080**	2.202	2.033	2.205	2.052	2.162	1.991	2.169	**1.973**	2.099	2.020	2.054	1.987
CAL	1.825	1.868	1.738	1.750	1.713	**1.714**	1.735	1.783	1.781	**1.582**	1.717	1.745	1.530	1.816
CON	20.50	24.76	12.00	**17.08**	14.64	21.95	7.639	**16.38**	7.480	16.74	16.77	25.93	9.897	19.67
STP	0.330	0.371	0.263	**0.299**	0.298	0.360	0.247	0.295	0.270	0.321	0.206	**0.279**	0.197	0.290
WAN	1.166	0.835	0.984	**0.729**	0.656	0.850	0.900	0.900	0.363	**0.834**	0.431	0.895	0.483	0.951
WIZ	0.675	0.724	0.637	**0.697**	0.661	0.709	0.595	0.700	0.506	**0.669**	0.658	0.746	0.615	0.677
MV	0.082	0.082	0.031	**0.031**	0.054	0.054	0.223	0.223	0.213	0.221	0.052	0.052	0.049	**0.048**
FOR	1095	4066	974.45	**3018**	1201	4021	1156	4053	1036	**2198**	1174	4079	1185	4080
MOR	0.010	0.012	0.008	**0.009**	0.010	0.010	0.013	0.010	0.004	**0.004**	0.007	0.008	0.008	0.008
TRE	0.022	0.036	0.022	**0.032**	0.024	0.036	0.019	0.033	0.015	**0.025**	0.016	0.029	0.017	0.028
BAS	1.568	**2.796**	1.578	2.827	2.349	2.804	1.684	2.748	1.790	**2.521**	1.832	2.755	1.510	2.828
HOU	7.954	8.279	6.421	**6.990**	8.145	8.211	6.741	6.987	6.197	6.861	5.979	**6.729**	5.677	6.803
ELV	2.457	**2.519**	2.452	2.554	2.568	2.706	2.136	2.220	1.957	**1.987**	2.213	2.341	2.292	2.327
CA	3.641	3.804	3.600	**3.769**	3.743	3.964	3.578	3.752	3.307	3.491	3.273	3.510	3.354	**3.432**
POLE	45.96	47.51	34.79	**36.16**	47.94	49.65	9.459	9.873	7.587	**8.788**	13.03	16.90	9.53	12.46
PUM	0.357	0.344	0.339	**0.326**	0.377	0.372	0.214	0.202	0.217	**0.157**	0.152	0.215	0.339	0.302
AIL	1.281	**1.342**	1.307	1.354	1.291	1.342	1.252	**1.309**	1.258	1.317	1.273	1.324	1.297	1.359

As can be observed from Table 5, when increasing the complexity of the FNNR-M (increasing the number of nodes in the consequent layer), the regression performance cannot be significantly improved. When the number of nodes in the consequent layer is increased to seven, the minimum error can only be achieved on CAL dataset. For the FNNR-F, when the complexity is reduced (reducing the number of fully connected layers), the regression performance of the model is greatly affected. When the number of fully connected layers is reduced to one, the model can only achieve the minimum MSEs on three datasets. It is worth noting that when the complexity of the FNNR-F is increased, the performance does not get better. On the one hand, this is related to the phenomenon of vanishing gradient; when the number of fully connected layers increases, the gradient at the back of the network has difficulty being transmitted to the front layers, resulting in low learning efficiency of the parameters of the fuzzification layer and fuzzy logic layers. On the other hand, the conclusion can be drawn by the comparison of MSEs on training sets and test sets that when the number of fully connected layers increases, the risk of overfitting gradually increases.

By conducting ablation experiments on the rule type represented by the output layers, it can be concluded that neither FNNR-M nor FNNR-T can properly balance prediction accuracy and interpretability. Therefore, it is most appropriate to adopt TSK fuzzy rules in the FNNR model.

4.3.3. The Ablation of Alternate Training Strategy

As mentioned above, to avoid the problem that the model has difficulty converging due to the oscillation of fuzzy partition parameters and fuzzy logic layer parameters in the training process, a three-stage alternate training strategy is designed. To demonstrate the effectiveness of this strategy, an ablation analysis is performed.

The FNNR-T is trained using the alternate training strategy and normal training method on all 28 datasets, and the average test errors are recorded. To highlight the role of alternate training strategy, parameters of MFs and fuzzy partitions are fixed during the training, and the whole training process is only divided into two stages: the stage of joint training and the stage of the fixed fuzzy logic layer. It is found that for 24 of the 28 datasets, the average MSEs are lower when using the alternate training strategy. This indicates that the alternate training strategy can help to improve the prediction accuracy of the model. Experiments are also conducted on the other two models, and relevant experimental data and analyses are shown in Supplementary S5.

Figure 5 reveals the training loss of the FNNR-T on low-dimensional datasets MPG6 and DEE and high-dimensional datasets PUM and ELV when the alternate training strategy and normal training method are adopted. For intuition, 5000 cycles are trained on low-dimensional datasets and 10,000 cycles are trained on high-dimensional datasets. It can be observed that the loss gradually diverges in the later training period and cannot converge when adopting the normal training method, while the alternate training strategy can help the loss gradually stabilize and finally converge.

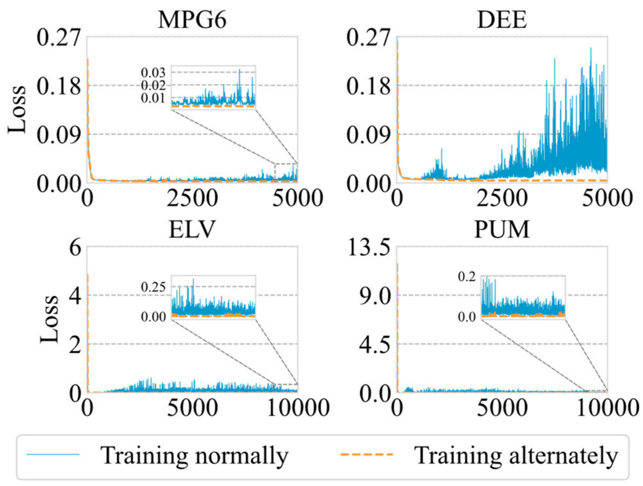

Figure 5. The training loss of the FNNR-T on MPG6, DEE, ELV, and PUM datasets.

4.4. Parameter Analysis

To understand the influence of regularization coefficients on the training and regression performance of the model, parameter analyses on λ_1 and λ_2 are carried out in this chapter.

4.4.1. The Parameter Analysis on the Regularization Coefficient λ_1

Firstly, the regularization coefficient λ_1 is analyzed. As mentioned above, λ_1 is the regularization coefficient to measure the complementarity of fuzzy sets. The larger λ_1 is, the stronger the complementarity of fuzzy sets will be, the clearer the semantics of fuzzy sets will be, and the stronger the interpretability will be. Figure 6 illustrates the average prediction errors of the FNNR-T under different values of λ_1 in high-dimensional datasets. Similarly, to clearly show the influence of regularization coefficients, fuzzy partition parameters are fixed in the training process.

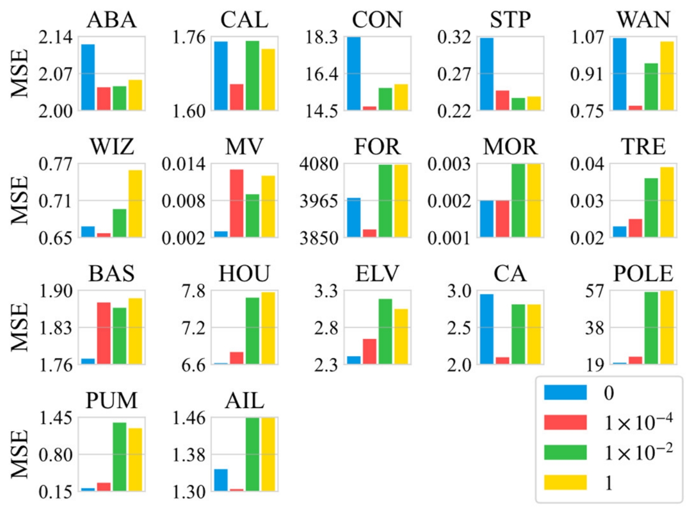

Figure 6. The average MSE of the model under different values of λ_1 on high-dimensional datasets.

As can be observed from Figure 6, when $\lambda_1 = 0$ and $\lambda_1 = 1e-4$, the model gets the minimum MSEs on eight datasets; under the circumstances of $\lambda_1 = 1$ and $\lambda_1 = 1e-2$, the model achieves the minimum error on only one dataset. This indicates that with the increase of λ_1, when it rises to a threshold, the regression accuracy of the model will decrease gradually. This is also in line with our subjective feelings: with the increase of the regularization coefficient, the model pays too much attention to the complementarity of fuzzy sets during training, which leads to a decrease in accuracy.

Figure 7 shows the fuzzy sets that the model eventually learns under the four values of λ_1 in the CON dataset, respectively. It can be observed that the interpretability of fuzzy sets is poor when $\lambda_1 = 0$. Some areas are covered repeatedly by a few fuzzy sets at the same time, so the semantics is peculiarly fuzzy. For example, when x_1 is near 0.8 with $\lambda_1 = 0$, the firing strengths of three fuzzy sets are very high, which is not intuitive. In addition, there are several fuzzy sets whose shapes are "sharp", such as the second fuzzy set of x_2 when $\lambda_1 = 0$, which is also poorly interpretable. With the increase of λ_1, the fuzzy sets tend to be uniform, and the interpretability is enhanced gradually.

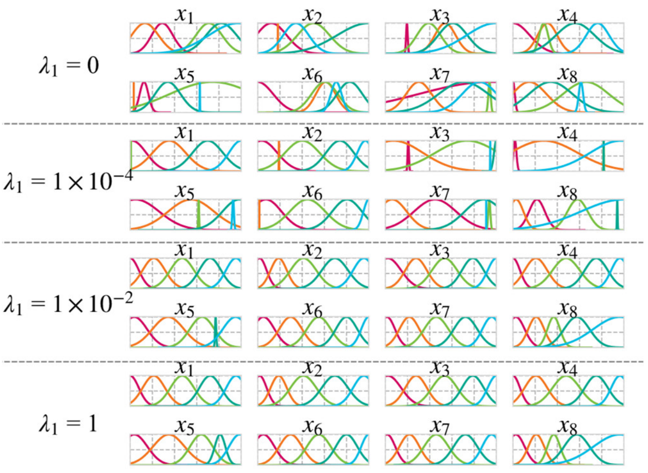

Figure 7. The fuzzy sets of each feature under different values of λ_1 in the CON dataset.

4.4.2. The Parameter Analysis on the Regularization Coefficient λ_2

The regularization coefficient λ_2 is analyzed below. λ_2 is the regularization coefficient to control the number of fuzzy sets of features. The larger λ_2 is, the fewer fuzzy sets there are, and the more explainable the model is. Figure 8 reveals the average prediction errors of the FNNR-T under different values of λ_2 on high-dimensional datasets. For the sake of fairness, λ_1 is set to 1×10^{-4}.

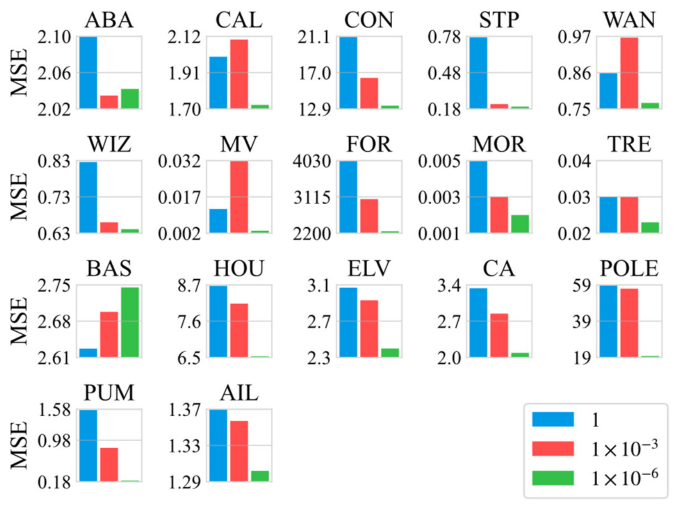

Figure 8. The average MSE of the model under different values of λ_2 on high-dimensional datasets.

As can be seen from Figure 8, when $\lambda_2 = 1e-6$, the model has the best regression performance, and it can reach the minimum errors on 13 of 17 datasets. When $\lambda_2 = 1$, the regression performance of the model is poor, and it can only reach the minimum MSE on one dataset. Compared with $\lambda_2 = 1e-6$, the average errors under $\lambda_2 = 1$ increase by more than 20% on 11 datasets. When $\lambda_2 = 1e-3$, the regression performance of the model is middle-ranking, and the prediction errors are minimized on two datasets. Compared with $\lambda_2 = 1e-6$, the errors increase by more than 20% on seven datasets under $\lambda_2 = 1e-3$. The results observed above are in line with our subjective feelings: With the increase of λ_2, the number of fuzzy partitions decreases continuously. Therefore, the remaining fuzzy sets are hard to reasonably divide in the input space, and the accuracy of the model is seriously affected.

Figure 9 illustrates the fuzzy sets finally learned by the model using different values of λ_2 on the STP dataset. From the figure, when the regularization coefficient is moderate, inappropriate fuzzy sets will be discarded, and the number of fuzzy sets used for each feature is well optimized. When the regularization coefficient is too large, many fuzzy sets are discarded, and the remaining fuzzy sets are difficult to partition in the input space reasonably and efficiently.

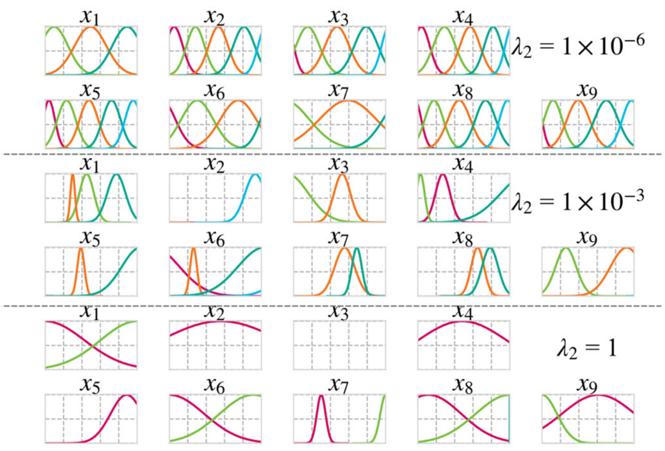

Figure 9. The fuzzy sets of each feature under different values of λ_2 in the STP dataset.

4.5. *The Interpretability of the FNNR*

The proposed FNNR has good interpretability, and TSK fuzzy rules can be directly extracted from the trained FNNR. The course of rule extraction is very simple: one fuzzy rule can be extracted if the node in the fuzzy logic layers whose parameter is one is found.

To visually illustrate the interpretability of the model, Figure 10 and Table 6 show the fuzzy rules extracted from the model in datasets ELE1 and MPG8, respectively, where L, M, and H are the names of fuzzy sets when the fuzzy partition number is three, and L, ML, M, MM, and H are the names of fuzzy sets when the fuzzy partition number is five. b is the constant coefficient of the consequent. As can be seen, whether the feature number is small or large, the model can achieve low regression error with a small number of rules.

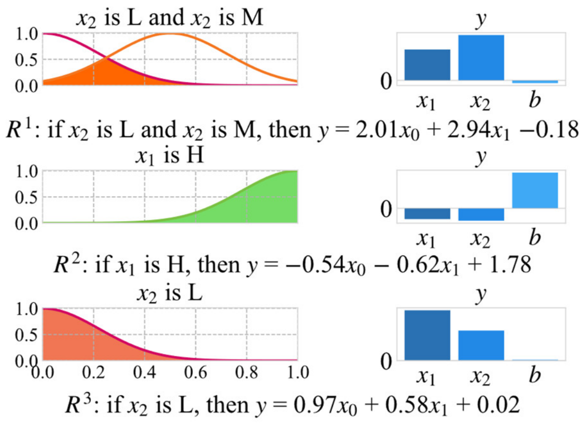

Figure 10. The fuzzy rules used by the FNNR on the dataset ELE1.

Table 6. The fuzzy rules used by the FNNR on the dataset MPG8.

No.		Rules
1	Antecedent	$x_1(L) \mid x_3(M) \mid x_3(MM) \mid x_4(M) \mid x_6(L) \mid x_6(MM) \mid x_7(M)$
	Consequent	$[0.02, 0.23, -0.24, -0.67, -0.15, 0.3, -0.14, 0.51]$
2	Antecedent	$x_3(M) \mid x_6(MM) \mid x_7(M)$
	Consequent	$[0.14, 0.02, -0.86, -1.15, 0.12, 0.60, 0.33, 0.93]$
3	Antecedent	$x_4(L) \& [x_3(M) \mid x_6(MM) \mid x_7(M)]$

	Consequent	$[0.23, 0.10, 0.12, -0.08, 0.57, 0.21, 0.04, -0.05]$
4	Antecedent	$x_2(ML) \mid x_4(H)$
	Consequent	$[0.36, 0.56, 0.45, 0.25, 0.21, 0.37, 0.20, 0.23]$
5	Antecedent	$x_4(L)$
	Consequent	$[0.37, 0.14, 0.50, 0.46, 0.28, 0.22, -0.17, 0.29]$
6	Antecedent	$[x_2(ML) \& x_4(H)] \& [x_1(H) \mid x_3(MM) \mid x_4(M) \mid x_6(L)]$
	Consequent	$[-0.35, -0.04, -0.17, -0.41, 0.16, 0.39, 0.16, 0.10]$
7	Antecedent	$x_1(H) \mid x_3(MM) \mid x_4(M) \mid x_6(L)$
	Consequent	$[0.01, 0.22, -0.19, -0.16, 0.01, -0.01, 0.37, 0.31]$

5. Conclusions and Future Work

A novel explainable fuzzy neural network regression method (FNNR) is proposed in this paper. To solve the problem of rule explosion on high-dimensional datasets, the symmetrical structure and corresponding parameter transformation method are used to learn the number of fuzzy rules and fuzzy partitions automatically. In addition, the structure identification and parameter identification of the model are considered. The number of fuzzy rules, the number of fuzzy partitions, the parameters of Gaussian MFs, and the consequent parameters are trained synergistically. On this basis, an alternate training strategy is designed to train different types of parameters to promote convergence. To further enhance the interpretability of the model, the regularized items are designed from fuzzy rule level and fuzzy partition level to guide the model to learn fuzzy rules with a simple structure and clear semantics. Experimental results on datasets with low and high dimensions show that the proposed model can achieve high test accuracy with good interpretability by comparing with some representative regression methods based on fuzzy rules and the classical regression models.

First, various uncertainties that may exist in the datasets, such as missing values, error values, noises, abnormal values, and so on, are not considered in this study. Future research could combine rough sets [46] and other technologies with fuzzy sets to deal with the above uncertainties. In addition, the type-I fuzzy sets used in this paper can also be extended to type-II fuzzy sets or interval fuzzy sets [47,48] to better deal with the uncertainty in the data. Secondly, in this study, Gaussian MFs are utilized to represent the fuzzy sets of all features. Considering different MFs and their combinations or using the shape of MFs as a learnable parameter is another possible future research direction. Finally, the gradient-based method is adopted in this study to train all parameters together as a whole. This method requires no intervention, but it is time-consuming in large datasets. Learning from some fast training methods, such as pseudoinverse [49] and heuristic greedy search [10], to conduct collaborative training for various parameters is also a potential future research topic.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/sym15091711/s1.

Author Contributions: Validation, W.H.; Writing—original draft, K.Z.; Writing—review & editing, X.Y.; Supervision, T.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Defense Industrial Technology Development Program] grant number [JCKY2020601B018]

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Das, R.; Sen, S.; Maulik, U. A survey on fuzzy deep neural networks. *ACM Comput. Surv.* **2020**, *53*, 1–25. https://doi.org/10.1145/3369798.
2. Kerk, Y.W.; Tay, K.M.; Lim, C.P. Monotone Fuzzy Rule Interpolation for Practical Modeling of the Zero-Order TSK Fuzzy Inference System. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 1248–1259. https://doi.org/10.1109/TFUZZ.2021.3057239.
3. Li, G.; Peng, C.; Xie, X.; Xie, S. On Stability and Stabilization of T–S Fuzzy Systems with Time-Varying Delays via Quadratic Fuzzy Lyapunov Matrix. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 3762–3773. https://doi.org/10.1109/TFUZZ.2021.3128062.
4. Arrieta, A.B.; Díaz-Rodríguez, N.; Del Ser, J.; Bennetot, A.; Tabik, S.; Barbado, A.; García, S.; Gil-López, S.; Molina, D.; Benjamins, R.; et al. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Inf. Fusion* **2020**, *58*, 82–115.
5. De Campos Souza, P.V. Fuzzy neural networks and neuro-fuzzy networks: A review the main techniques and applications used in the literature. *Appl. Soft Comput.* **2020**, *92*, 106275.
6. Škrjanc, I.; Iglesias, J.A.; Sanchis, A.; Leite, D.; Lughofer, E.; Gomide, F. Evolving fuzzy and neuro-fuzzy approaches in clustering, regression, identification, and classification: A survey. *Inf. Sci.* **2019**, *490*, 344–368.
7. Deng, Y.; Ren, Z.; Kong, Y.; Bao, F.; Dai, Q. A hierarchical fused fuzzy deep neural network for data classification. *IEEE Trans. Fuzzy Syst.* **2017**, *25*, 1006–1012. https://doi.org/10.1109/TFUZZ.2016.2574915.
8. Zhang, Y.; Liu, Y.; Li, Q.; Tiwari, P.; Wang, B.; Li, Y.; Pandey, H.M.; Zhang, P.; Song, D. CFN: A complex-valued fuzzy network for sarcasm detection in conversations. *IEEE Trans. Fuzzy Syst.* **2021**, *29*, 3696–3710. https://doi.org/10.1109/TFUZZ.2021.3072492.
9. Yang, C.H.; Moi, S.H.; Hou, M.F.; Chuang, L.Y.; Lin, Y.D. Applications of deep learning and fuzzy systems to detect cancer mortality in next-generation genomic data. *IEEE Trans. Fuzzy Syst.* **2021**, *29*, 3833–3844. https://doi.org/10.1109/TFUZZ.2020.3028909.
10. Wang, N.; Pedrycz, W.; Yao, W.; Chen, X.; Zhao, Y. Disjunctive Fuzzy Neural Networks: A New Splitting-Based Approach to Designing a T–S Fuzzy Model. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 370–381. https://doi.org/10.1109/TFUZZ.2020.3039371.
11. Jang, J.-S.R. ANFIS: Adaptive-network-based fuzzy inference system. *IEEE Trans. Syst. Man Cybern.* **1993**, *23*, 665–685. https://doi.org/10.1109/21.256541.
12. Eyoh, I.; John, R.; De Maere, G. Interval type-2 A-intuitionistic fuzzy logic for regression problems. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 2396–2408. https://doi.org/10.1109/TFUZZ.2017.2775599.
13. Park, S.; Lee, S.J.; Weiss, E.; Motai, Y. Intra- and inter-fractional variation prediction of lung tumors using fuzzy deep learning. *IEEE J. Transl. Eng. Health Med.* **2016**, *4*, 1–12. https://doi.org/10.1109/JTEHM.2016.2516005.
14. Xue, G.; Wang, J.; Yuan, B.; Dai, C. DG-ALETSK: A High-Dimensional Fuzzy Approach with Simultaneous Feature Selection and Rule Extraction. In *IEEE Transactions on Fuzzy Systems*; IEEE: New York, NY, USA, 2023. https://doi.org/10.1109/TFUZZ.2023.3270445.
15. Wu, D.; Yuan, Y.; Huang, J.; Tan, Y. Optimize TSK fuzzy systems for big data regression problems: Mini-batch gradient descent with regularization, droprule and adabound (MBGD-RDA). *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 1003–1015.
16. Fidan, S.; Karasulu, B. Clustering Methods Comparison for Optimization of Adaptive Neural Fuzzy Inference System. In Proceedings of the 2022 30th Signal Processing and Communications Applications Conference (SIU), Safranbolu, Turkey, 15–18 May 2022; pp. 1–4. https://doi.org/10.1109/SIU55565.2022.9864902.
17. Souza PV, D.C.; Guimares, A.J.; Rezende, T.S.; Araujo, V.S.; Araujo VJ, S.; Batista, L.O. Bayesian fuzzy clustering neural network for regression problems. In Proceedings of the 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), Bari, Italy, 6–9 October 2019; pp. 1492–1499. https://doi.org/10.1109/SMC.2019.8914212.
18. Huang, W.; Oh, S.-K.; Pedrycz, W. Fuzzy wavelet polynomial neural networks: Analysis and design. *IEEE Trans. Fuzzy Syst.* **2017**, *25*, 1329–1341.
19. Palconit MG, B.; Conception, R.S., II; Alejandrino, J.D.; Nuñez, W.A.; Bandala, A.A.; Dadios, E.P. Comparative ANFIS Models for Stochastic On-road Vehicle CO_2 Emission using Grid Partitioning, Subtractive, and Fuzzy C-means Clustering. In Proceedings of the 2021 IEEE 9th Region 10 Humanitarian Technology Conference (R10-HTC), Bangalore, India, 30 September–2 October 2021; pp. 1–6. https://doi.org/10.1109/R10-HTC53172.2021.9641644.
20. Ouyang, C.S.; Kao, T.C.; Cheng, Y.Y.; Wu, C.H.; Tsai, C.H.; Wu, M.W. An improved fuzzy extreme learning machine for classification and regression. In Proceedings of the International Conference on Cybernetics, Robotics and Control (CRC), Hong Kong, China, 19–21 August 2016; pp. 91–94.
21. Zhao, K.; Dai, Y.; Jia, Z.; Ji, Y. General Fuzzy C-Means Clustering Strategy: Using Objective Function to Control Fuzziness of Clustering Results. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 3601–3616. https://doi.org/10.1109/TFUZZ.2021.3119240.
22. Dey, S.; Dam, T. Rainfall-runoff prediction using a Gustafson-Kessel clustering based Takagi-Sugeno Fuzzy model. In Proceedings of the 2021 IEEE Symposium Series on Computational Intelligence (SSCI), Orlando, FL, USA, 5–7 December 2021; pp. 1–8. https://doi.org/10.1109/SSCI50451.2021.9660037.
23. Deng, Z.H.; Choi, K.S.; Wang, S.T. Scalable TSK fuzzy modeling for very large datasets using minimal-enclosing-ball approximation. *IEEE Trans. Fuzzy Syst.* **2011**, *19*, 210–226.
24. Leski, J.M. SparseFIS: Data-driven learning of fuzzy systems with sparsity constraints. *IEEE Trans. Fuzzy Syst.* **2010**, *18*, 396–411.
25. Yi-Zhang, J.; Zhao-Hong, D.; Shi-Tong, W. Mamdani-Larsen type transfer learning fuzzy system. *Acta Autom. Sin.* **2012**, *38*, 1393–1409.

26. Pal, N.R.; Mudi, R.; Pal, K. Rule extraction through exploratory data analysis for self-tuning fuzzy controllers. *Int. J. Fuzzy Syst.* **2004**, *6*, 71–80.
27. Zhang, Y.; Ishibuchi, H.; Wang, S. Deep Takagi–Sugeno–Kang fuzzy classifier with shared linguistic fuzzy rules. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 1535–1549. https://doi.org/10.1109/TFUZZ.2017.2729507.
28. de Jesus Rubio, J. SOFMLS: Online self-organizing fuzzy modified least-squares network. *IEEE Trans. Fuzzy Syst.* **2009**, *17*, 1296–1309.
29. Xue, G.; Chang, Q.; Wang, J.; Zhang, K.; Pal, N.R. An Adaptive Neuro-Fuzzy System with Integrated Feature Selection and Rule Extraction for High-Dimensional Classification Problems. *IEEE Trans. Fuzzy Syst.* **2023**, *31*, 2167–2181. https://doi.org/10.1109/TFUZZ.2022.3220950.
30. Wang, D.; Zeng, X.-J.; Keane, J.A. Hierarchical hybrid fuzzy neural networks for approximation with mixed input variables. *Neurocomputing* **2007**, *70*, 3019–3033.
31. Trillo, J.R.; Fernandez, A.; Herrera, F. HFER: Promoting Explainability in Fuzzy Systems via Hierarchical Fuzzy Exception Rules. In Proceedings of the 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Glasgow, UK, 19–24 July 2020; pp. 1–8. https://doi.org/10.1109/FUZZ48607.2020.9177575.
32. Chen, J. Adaptive Fuzzy Neural Network Control Based on Genetic Algorithm. In Proceedings of the 2021 13th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), Beihai, China, 16–17 January 2021; pp. 393–396. https://doi.org/10.1109/ICMTMA52658.2021.00091.
33. Kumari, N.; Gill, A.; Singh, M. Two-Area Power System Load Frequency Regulation Using ANFIS and Genetic Algorithm. In Proceedings of the 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 26–28 May 2023; pp. 1–7. https://doi.org/10.1109/INCET57972.2023.10170037.
34. Tung, S.; Quek, C.; Guan, C. eT2fifis: An evolving type-2 neural fuzzy inference system. *Inf. Sci.* **2013**, *220*, 124–148.
35. Gacto, M.J.; Galende, M.; Alcalá, R.; Herrera, F. METSK-HDe: A multiobjective evolutionary algorithm to learn accurate TSK-fuzzy systems in high-dimensional and large-scale regression problems. *Inf. Sci.* **2014**, *276*, 63–79.
36. Aghaeipoor, F.; Javidi, M.M. MOKBL+MOMs: An interpretable multi-objective evolutionary fuzzy system for learning high-dimensional regression data. *Inf. Sci.* **2019**, *496*, 1–24.
37. Zhang, K.; Hao, W.N.; Yu, X.H.; Chen, G.; Yu, K. A fuzzy neural network classifier and its dual network for adaptive learning of structure and parameters. *Int. J. Fuzzy Syst.* **2023**, *25*, 1034–1054. https://doi.org/10.1007/s40815-022-01392-y.
38. Gacto, M.J.; Alcalá, R.; Herrera, F. Interpretability of linguistic fuzzy rule-based systems: An overview of interpretability measures. *Inf. Sci.* **2011**, *181*, 4340–4360.
39. Alcal-Fdez, J.; Fernndez, A.; Luengo, J.; Derrac, J.; Garca, S.; Snchez, L.; Herrera, F. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *J. Mult.-Valued Log. Soft Comput.* **2011**, *17*, 255–287.
40. Quinlan, J.R. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106.
41. Rodrıguez-Fdez, I.; Mucientes, M.; Bugarın, A. FRULER: Fuzzy rule learning through evolution for regression. *Inf. Sci.* **2016**, *354*, 1–18.
42. Cozar, J.; delaOssa, L.; Gamez, J.A. Learning compact zero-order TSK fuzzy rule-based systems for high-dimensional problems using an Apriori + local search approach. *Inf. Sci.* **2018**, *433*, 1–16.
43. Friedman, M. A comparison of alternative tests of significance for the problem of m rankings. *Ann. Math. Stat.* **1939**, *11*, 86–92. https://doi.org/10.1214/aoms/1177731944.
44. Dunn, O.J. Multiple comparisons among means. *J. Am. Stat. Assoc.* **1961**, *56*, 52–64. https://doi.org/10.1080/01621459.1961.10482090.
45. Zheng, X.-J.; Singh, M.G. Approximation accuracy analysis of fuzzy systems with the center-average defuzzifier. In Proceedings of the 1995 IEEE International Conference on Fuzzy Systems, Yokohama, Japan, 20–24 March 1995; Volume 1, 109–116. https://doi.org/10.1109/FUZZY.1995.409668.
46. Pawlak, Z. *Rough Sets: Theoretical Aspects of Reasoning about Data*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012; Volume 9.
47. Mendel, J.M. *Uncertain Rule-Based Fuzzy Systems: Introduction and New Directions*; 2nd ed.; Springer: Cham, Switzerland, 2017.
48. Wu, D. On the fundamental differences between interval type-2 and type-1 fuzzy logic controllers. *IEEE Trans. Fuzzy Syst.* **2012**, *20*, 832–848.
49. Feng, S.; Chen, C.L.P. Fuzzy broad learning system: A novel neuro-fuzzy model for regression and classification. *IEEE Trans. Cybern.* **2020**, *50*, 414–424. https://doi.org/10.1109/TCYB.2018.2857815.