Article

Dynamic System Identification and Prediction Using a Self-Evolving Takagi–Sugeno–Kang-Type Fuzzy CMAC Network

Cheng-Jian Lin ^{1,*} , Cheng-Hsien Lin ² and Jyun-Yu Jhang ¹

¹ Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 411, Taiwan; o800920@gmail.com

² Department of Electrical Engineering, National Chung Hsing University, Taichung 402, Taiwan; andy39722@gmail.com

* Correspondence: cjlin@ncut.edu.tw

Received: 20 February 2020; Accepted: 9 April 2020; Published: 10 April 2020

Abstract: This study proposes a Self-evolving Takagi-Sugeno-Kang-type Fuzzy Cerebellar Model Articulation Controller (STFCMAC) for solving identification and prediction problems. The proposed STFCMAC model uses the hypercube firing strength for generating external loops and internal feedback. A differentiable Gaussian function is used in the fuzzy hypercube cell of the proposed model, and a linear combination function of the model inputs is used as the output of the proposed model. The learning process of the STFCMAC is initiated using an empty hypercube base. Fuzzy hypercube cells are generated through structure learning, and the related parameters are adjusted by a gradient descent algorithm. The proposed STFCMAC network has some advantages that are summarized as follows: (1) the model automatically selects the parameters of the memory structure, (2) it requires few fuzzy hypercube cells, and (3) it performs identification and prediction adaptively and effectively.

Keywords: prediction; identification; fuzzy model; cerebellar model articulation controller; recurrent network

1. Introduction

During the past decade, neural networks (NNs) have been widely used in dynamic system applications, such as control, identification, prediction, and signal processing [1–4]. NNs exhibit the advantages of effective function approximation, adaptive learning, generalization abilities, and computation parallelism. However, their disadvantages include computational complexity and slow convergence.

Based on the neurophysiological properties of the human cerebellum, Albus [5,6] presented a cerebellar model articulation controller (CMAC). The associative memory is constructed using the overlapping receptive fields. A corresponding relationship is present between the input and output of the mapping. The CMAC has a highly standardized computational structure, fast network learning, local generalization, and fast convergence [5–9]. However, because of its constant response and quantified receptive fields, the approximation capacity of the CMAC model is limited. In other words, the receptive fields are fixed in the conventional CMAC model.

To overcome the aforementioned problems, several studies have proposed improving the performance of the CMAC model by using differentiable cells with fuzzy boundaries [10–12]. Because the CMAC develops differentiable functions through learning, it exhibits structural flexibility in the local region. Several researchers [13–16] have combined the CMAC and fuzzy logic linguistic representation to solve problems that include processing uncertain and nonlinear problems. Sim et al. [13] introduced Bayesian Ying–Yang learning to optimize an FCMAC. Forward training and backward running phases were adopted for input–output discrimination. Wu et al. [14] proposed an adaptive mechanism for

FCMAC learning. Wu [15] investigated the trajectory tracking control of wheeled mobile robots by using an FCMAC. Compared with the CMAC, the FCMAC uses fuzzy membership functions to model the problem. As a result, the FCMAC is highly intuitive and easy to understand.

Several methods [17,18] have employed different strategies to enhance the efficiency of CMACs in previous studies. Zeng and Keane [17] combined Kolmogorov's theorem and hierarchical fuzzy systems to promote the universal approximation property. To improve the function approximation ability, Lee et al. [18] proposed a parametric FCMAC (PFCMAC) that is a hybrid of a Takagi–Sugeno–Kang (TSK)-type fuzzy inference system [19] and a CMAC network. The PFCMAC can approximate continuous functions and minimize the number of hypercube cells. These models use a feedforward structure and therefore cause instability problems due to the local nature of hypercube cells. At the same time, this problem also occurs with overtraining in static or dynamic systems [20,21]. The dynamic systems involve correlation between the input and output; several types of recurrent technique involve the use of this mechanism for NNs and fuzzy systems [22–26]. The relative position of delay units can be adjusted to achieve precise control and enable the accurate approximation of actual values. Therefore, recurrent networks can overcome the disadvantages of feedforward networks. There are two types of recurrent structure. In one recurrent structure, global feedback is used in fuzzy NNs (FNNs) [27–30]. The other structure uses internal state variables as local recurrent feedback loops [27–30]. However, the aforementioned structures are incapable of mapping.

In this study, we extend our previous study [18] by developing a Self-evolving Takagi–Sugeno–Kang-type Fuzzy Cerebellar Model Articulation Controller (STFCMAC) model. The interactively recurrent structure of the proposed model provides a strong search ability for local and global solutions. Global feedback is obtained from itself and other fuzzy hypercube cells. However, local feedback is insufficient to represent all necessary information. In other words, a fuzzy hypercube cell receives feedback from itself only. Several studies have considered past states in recurrent structures without referring to current states and thus obtained insufficient information. The three major contributions of the proposed STFCMAC are summarized as follows: (1) the model automatically selects the parameters of the memory structure, (2) it requires few fuzzy hypercube cells, and (3) it performs identification and prediction adaptively and effectively.

Several types of simple CMAC are introduced in Section 2. The STFCMAC model is proposed in Section 3. Section 4 presents the learning algorithm of the STFCMAC. Section 5 illustrates the experimental results for identifying nonlinear dynamic systems and predicting time series. Finally, conclusions are provided in Section 6.

2. Review Fuzzy CMAC (FCMAC) Models

Fuzzy CMAC Model

The fuzzy CMAC model is similar to the traditional CMAC model but uses two main mappings $S(x)$ and $P(s)$ based on fuzzy operations to approximate the nonlinear function $y = f(x)$. The FCMAC model is shown in Figure 1. In this case, a Gaussian function is used to model the receptive field basis function and add fuzzy weights to the result. The five layers are described as follows. Each input variable x_i in Layer 1 is quantized into discrete regions or elements. Several elements in Layer 1 can accumulate to form a block. Each component performs a Gaussian basis function. Layer 2 is an associated memory space and corresponds to a linguistic variable in each function represented by a membership function. Layer 2 can be regarded as fuzzifying the input variables. Layer 3 is the receptive field space or fuzzy hypercube. Each node implements a fuzzy operation and obtains the firing intensity s . The fuzzy weights in Layer 4 are inferred to generate a partial fired fuzzy output by its fuzzy hypercube selection vector as the matching degree of inputs. In Layer 5, a centroid of area approach is adopted to obtain the model output.

Figure 1. The structure of the Fuzzy Cerebellar Model Articulation Controller (FCMAC) model.

3. Proposed STFCMAC Model

Structural learning and parametric learning in the STFCMAC model—an extension of our previous study [18]—is described in this section. The recurrent structure in the STFCMAC model uses an interactive feedback mechanism that captures key information from other hypercube units and can be combined with TSK-type linear functions for a better solution. The proposed STFCMAC model is different from that proposed by Lin et al. [31], which employs a fuzzy neural network structure; our model employs a TSK-type fuzzy CMAC structure.

The proposed STFCMAC model employs a recurrent feedback mechanism in the temporal layer and a linear combination function in the consequent part to ensure high performance of the network. Figure 2 displays the proposed six-layered STFCMAC structure. The proposed STFCMAC realizes a similar fuzzy if-then rule:

IF x_i is A_{1j} and x_2 is A_{2j} ... and x_i is A_{ij} ... and x_{N_D} is A_{N_Dj}

THEN $\hat{y}_j = \sum_{j=1}^{O_j^{(4)}} (c_{0j} + \sum_{i=1}^{N_D} c_{ij}x_i)$

where x_i and $\hat{y}_j$ represent the input and output variables, A_{ij} is the linguistic term of the preconditioning of the Gaussian membership function, N_D is the number of input dimensions, $O_k^{(4)}$ is the output of interactive feedback, and $c_{0j} + \sum_{i=1}^{N_D} c_{ij}x_i$ is the linear combination function of the input.

Figure 2. The proposed Self-evolving Takagi-Sugeno-Kang-type Fuzzy Cerebellar Model Articulation Controller (STFCMAC) model.

The structure of the STFCMAC model is illustrated as follows:

Layer 1: Each node in the layer directly transfers the input value to the next layer.

$$O_i^{(1)} = I_i^{(1)}, \text{ and } I_i^{(1)} = x_i \quad (1)$$

Layer 2: In this layer, each fuzzy set A_{ij} performs a fuzzification operation.

$$O_{ij}^{(2)} = \exp\left(-\frac{[I_i^{(2)} - m_{ij}]^2}{\sigma_{ij}^2}\right), \text{ and } I_i^{(2)} = O_i^{(1)} \quad (2)$$

Layer 3: The firing strength α_j is computed by the product operation.

$$O_{ij}^{(2)} = \exp\left(-\frac{[I_i^{(2)} - m_{ij}]^2}{\sigma_{ij}^2}\right), \text{ and } I_i^{(2)} = O_i^{(1)} \quad (3)$$

where $\prod_i I_{ij}^{(3)}$ denotes the firing strength.

Layer 4: In this layer, the recurrent node performs internal feedback and external feedback loops. The output depends on both the previous and current firing strengths.

$$O_j^{(4)} = \sum_{k=1} \left(\lambda_{kj}^q \cdot O_k^{(4)}(t-1) \right) + \left(1 - \gamma_j^q \right) I_j^{(4)}, \text{ and } I_j^{(4)} = O_j^{(3)} \quad (4)$$

where $\gamma_j^q = \sum_{k=1}^M \lambda_{kj}^q$ and $\lambda_{kj}^q = \frac{R_{kj}^q}{M} (0 \leq R_{kj}^q \leq 1)$ represent the rule.

Layer 5: Each node in this layer combines a linear combination function of inputs and the corresponding feedback loop output from Layer 4.

$$O_j^{(5)} = O_j^{(4)} (c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i) \quad (5)$$

where c_{0j} and c_{ij} are the constant values and N_D represents the number of the input dimensions.

Layer 6: The centroid of area method is adopted for performing the defuzzification operation in this layer. The actual output y is described in the following:

$$y = \frac{\sum_{j=1}^N O_j^{(4)} (c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i)}{\sum_{j=1}^N O_j^{(4)}} \quad (6)$$

4. Learning Algorithm for Proposed STFCMAC Model

The proposed supervised learning algorithm comprises both structure and parameter learning schemes. A flowchart of the structure and parameter learning schemes is shown in Figure 3. Firstly, the STFCMAC model has no fuzzy hypercubes. The degree measure determines the self-partition of the input space in the structure learning scheme. Secondly, the parametric learning scheme uses a backpropagation algorithm to adjust parameters in the STFCMAC model for minimizing a given cost function.

4.1. Structure Learning Scheme

A new fuzzy hypercube is generated in the structural learning scheme. The firing strength α_j in Layer 3 is used as the degree measure after a product operation:

$$y = \frac{\sum_{j=1}^N O_j^{(4)} (c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i)}{\sum_{j=1}^N O_j^{(4)}} \quad (7)$$

The maximum degree measure S_{max} is determined as

$$S_{max} = \max_{1 \leq j \leq N} S_j \quad (8)$$

where N is the current number of fuzzy hypercube cells. Here, the parameter of a prespecified threshold $\bar{S}$ is defined. If $S_{max} \leq \bar{S}$, a new fuzzy hypercube cell is generated. Otherwise, no new fuzzy hypercube cell is generated. To avoid increasing the size of the STFCMAC model, the prespecified threshold should be reduced during the learning process. The selected threshold value will be subjective to different problems. That is, it depends on user experience or trial and error.

Figure 3. A flowchart of the proposed structure and parameter learning algorithm.

4.2. Parameter Learning Scheme

The backpropagation algorithm is used in the parameter learning scheme to adjust the parameters in the STFCMAC model. For ease of explanation, consider taking a single output as an example. The cost function $E(t)$ is defined as follows:

$$E(t) = \frac{1}{2} [y^d(t) - y(t)]^2 \quad (9)$$

where $y^d(t)$ and $y(t)$ are the desired and actual model outputs, respectively, at time t . The general backpropagation learning algorithm is written in the following:

$$P(t+1) = P(t) + \Delta P(t) = P(t) + \left[-\eta \frac{\partial E(t)}{\partial P(t)} \right] \quad (10)$$

where η is the learning rate and P denotes an adjustable parameter of the STFCMAC model. The adjustable parameters P are calculated using the gradient of error function $E(\cdot)$.

$$\frac{\partial E(t)}{\partial P} = e(t) \frac{\partial y(t)}{\partial P} \quad (11)$$

A recursive error term is generated in each layer by a chain rule to adjust the tunable parameters in the corresponding layer. Then, the parameters in the corresponding antecedent and consequent parts of the STFCMAC model are adjusted. The update rule for parameter c_{ij} can be exported as follows:

$$c_{ij}(t+1) = c_{ij}(t) + \Delta c_{ij}(t) \quad (12)$$

where

$$\Delta c_{ij}(t) = -\eta \cdot \frac{\partial E}{\partial c_{ij}} = -\eta \cdot \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial O_j^{(5)}} \cdot \frac{\partial O_j^{(5)}}{\partial c_{ij}} \quad (13)$$

The recurrent weight parameter λ_{kj}^q of each cell is updated based on the following equations:

$$\lambda_{kj}^q(t+1) = \lambda_{kj}^q(t) + \Delta \lambda_{kj}^q(t) \quad (14)$$

where

$$\Delta \lambda_{kj}^q(t) = -\eta \cdot \frac{\partial E}{\partial \lambda_{kj}^q} = -\eta \cdot \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial O_j^{(5)}} \cdot \frac{\partial O_j^{(5)}}{\partial O_j^{(4)}} \cdot \frac{\partial O_j^{(4)}}{\partial \lambda_{kj}^q} = -\eta \cdot \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial O_j^{(5)}} \cdot \frac{\partial O_j^{(5)}}{\partial O_j^{(4)}} \cdot \frac{\partial O_j^{(4)}}{\partial \lambda_{kj}^q} \quad (15)$$

where η is a real value belonging to $(0, 1)$, λ represents the learning rate, and e denotes the difference between the desired output and the model output.

The mean m_{ij} and variance σ_{ij} of the receptive field functions are updated in the following equations:

$$m_{ij}(t+1) = m_{ij}(t) + \Delta m_{ij}(t) \quad (16)$$

and

$$\sigma_{ij}(t+1) = \sigma_{ij}(t) + \Delta \sigma_{ij}(t) \quad (17)$$

where

$$\begin{aligned} \Delta m_{ij} &= -\eta \cdot \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial O_j^{(5)}} \cdot \frac{\partial O_j^{(5)}}{\partial O_j^{(4)}} \cdot \frac{\partial O_j^{(4)}}{\partial O_j^{(3)}} \cdot \frac{\partial O_j^{(3)}}{\partial O_j^{(2)}} \cdot \frac{\partial O_j^{(2)}}{\partial m_{ij}} \\ &= -\eta \cdot e \cdot \frac{(c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i) \sum_{j=1}^N O_j^{(4)} - \sum_{j=1}^N O_j^{(4)} (c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i)}{(\sum_j^N O_j^{(4)})^2} \cdot (1 - \gamma_j^q) \cdot S_j \cdot \frac{2(I_i^{(1)} - m_{ij})}{\sigma_{ij}^2} \end{aligned} \quad (18)$$

and

$$\begin{aligned} \Delta \sigma_{ij} &= -\eta \cdot \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial O_j^{(5)}} \cdot \frac{\partial O_j^{(5)}}{\partial O_j^{(4)}} \cdot \frac{\partial O_j^{(4)}}{\partial O_j^{(3)}} \cdot \frac{\partial O_j^{(3)}}{\partial O_j^{(2)}} \cdot \frac{\partial O_j^{(2)}}{\partial \sigma_{ij}} \\ &= -\eta \cdot e \cdot \frac{(c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i) \sum_{j=1}^N O_j^{(4)} - \sum_{j=1}^N O_j^{(4)} (c_{0j} + \sum_{i=1}^{N_D} c_{ij} x_i)}{(\sum_j^N O_j^{(4)})^2} \cdot (1 - \gamma_j^q) \cdot S_j \cdot \frac{2(I_i^{(1)} - m_{ij})^2}{\sigma_{ij}^3} \end{aligned} \quad (19)$$

All of the aforementioned formulas pertain to the case of a multiple-input, single-output system. For a multi-input and multi-output system, the cost function is rewritten in the following equations:

$$E = \frac{1}{2k} \sum_{k=1}^n (y_k^d(t) - y_k(t))^2 \quad (20)$$

where k is the number of outputs, and $k = 1, 2, \dots, n$.

5. Experimental Results

To illustrate the identification and prediction performances of the proposed STFCMAC model, three simulation examples, involving two dynamic system identification problems and a Mackey–Glass chaotic series prediction problem, are described in this section. In the two examples of dynamic system identification, we focus on comparing the performance of STFCMAC model with those of different recurrent fuzzy neural networks. In addition to the aforementioned comparison method, the different structural networks are also used to demonstrate the superiority of STFCMAC in Mackey–Glass chaotic series prediction.

5.1. Example 1: Identification of Nonlinear System

In this example, a nonlinear dynamic system is identified using the STFCMAC model. The difference equation of the nonlinear system is described as follows:

$$y_p(t+1) = f(y_p(t), y_p(t-1), y_p(t-2), x(t), x(t-1)) \quad (21)$$

where

$$f(p_1, p_2, p_3, p_4, p_5) = \frac{p_1 p_3 p_5 (p_3 - 1) p_4}{1 + p_2^2 + p_3^2} \quad (22)$$

The initial parameters are set as $\eta = 0.1$ and $\bar{S} = 0.0001$. The system uses the first two inputs and three previous outputs to get the output. In training the STFCMAC model, we use only ten epochs, and there are 900 time steps in each epoch. Similar to the inputs used in [29,32], the input is an *iid* uniform sequence over $(-2, 2)$ for about half of the 900 time steps and a sinusoid function, $1.05\sin(\pi t/45)$, given for the remaining time. There is no repetition on these 900 training data, i.e., we have different training sets for each epoch. The used time step in this paper is equal to 1. After training, three hypercube cells are generated. The testing input $x(t)$ is as follows:

$$x(t) = \begin{cases} \sin\left(\frac{\pi t}{25}\right), & t < 250 \\ 1.0, & 250 \leq t < 500 \\ -1.0, & 500 \leq t < 750 \\ 0.3 \sin\left(\frac{\pi t}{25}\right) + 0.1 \sin\left(\frac{\pi t}{32}\right) + 0.6 \sin\left(\frac{\pi t}{10}\right), & 750 \leq t < 1000 \end{cases} \quad (23)$$

To fairly compare the experimental results obtained, STFCMAC and other methods use the same number of training and test data and input variables. The performance of the STFCMAC model is compared with that of the self-organizing recurrent fuzzy CMAC model for dynamic system identification (RFCMAC) [24], the high-order recurrent neuro-fuzzy system (HO-RNFS) [28], the TSK-type recurrent fuzzy network (TRFN) [29], the wavelet recurrent fuzzy NN (WRFNN) [33], and the recurrent self-evolving NN with local feedback (RSEFNN-LF) [32]. The performance comparison of the various models in terms of dynamic system identification includes the fuzzy hypercube cells or fuzzy rules, number of parameters, training root-mean-square error (RMSE), and testing RMSE. The comparison results are presented in Table 1. Figure 4; Figure 5, respectively, display the identification results and errors between the real output and the output obtained when the STFCMAC model is used. In the third row of Table 1, the proposed STFCMAC requires fewer parameters than the other methods, except RFCMAC. The experimental results indicate that the proposed STFCMAC model exhibits better identification ability in RMSE than the other methods.

Table 1. A comparison of different identifiers in terms of dynamic system identification.

Models	Lin and Chin [33]	Theocharis [28]	Juang [29]	Juang et al. [32]	Chen [24]	Proposed STFCMAC
Fuzzy Rules/ Hypercubes	5	3	3	4	3	3
No. of Parameters	55	45	33	32	24	27
RMSE of Training Process	0.064	0.054	0.032	0.02	0.022	0.019
Training Time	9.18 s	13.93 s	192.27 s	13.78 s	10.53 s	12.37 s
RMSE of Testing Process	0.098	0.082	0.047	0.04	0.036	0.033

Figure 4. The results for the identification of a dynamic system obtained using the STFCMAC model. **Figure 5.** The errors in dynamic system identification between the real output and output obtained using the STFCMAC model.

In addition, two simulations, including the different training magnitude regions and the different input delays, are used to observe their effects on the proposed model. Firstly, the testing signal in Equation (23) is also used for this simulation. We adopt the different training magnitude regions to illustrate the effects. That is, the input is an *iid* uniform sequence on $(-2, 2)$, $(-1.6, 1.6)$, and $(-1.2, 1.2)$. Three different training magnitudes for about half of the 900 time steps and a sine function, $1.05\sin(\pi t/45)$ are used to generate the remaining 450 training inputs. Figure 6 presents the identification results for a dynamic system using the STFCMAC model with different training magnitude regions. The simulation results show that the training magnitude region is relatively narrow, from -1.2 to 1.2 , to obtain better identification results.

Secondly, the different input delay simulations are used to explore the relationship between the RMSE of testing and the input delay in the proposed model. In this simulation, input delays of 5 to 30 are used in the proposed model. Figure 7 illustrates the relationship between the testing RMSE and the time delay. In this figure, we find that if the degree of time delay is 30, the proposed model starts to perform very poorly.

Figure 6. The identification results for a dynamic system using the STFCMAC model with different training magnitude regions.

Figure 7. The relationship between the testing root-mean-square error (RMSE) and the time delay.

5.2. Example 2: System Identification of Longer Input Delays

In this example, a system identification of longer input delays is considered. The equation for identification is as follows:

$$y_p(t+1) = 0.72y_p(t) + 0.025y_p(t-1)x_1(t-1) + 0.01x_1^2(t-2) + 0.2x(t-3) \quad (24)$$

The plant system output is based on four previous inputs and two previous outputs. In the training procedure, 10 epochs are used, and each epoch comprises 900 time steps. The initial learning rate η is set at 0.15, and the decay threshold $\bar{S}$ is set at 0.0001 during the learning process. After training, three hypercube cells are generated. The testing signal used in Example 1 is also used for this example. For fair evaluation, the same parameters, such as the number of input variables, training data, and testing data, are used in

the STFCMAC model and the other models. Table 2 compares the results obtained by the STFCMAC model and other models [24,28,29,32,33]. Figures 8 and 9 respectively illustrate the identification results and errors between the real output and the output obtained using the STFCMAC model. The proposed TSK-type IRSFCMAC model outperforms the other network models.

Table 2. A comparison of different identifiers in terms of the dynamic system identification of longer input delays.

Models	Lin and Chin [33]	Theocharis [28]	Juang [29]	Juang et al. [32]	Chen [24]	Proposed STFCMAC
Fuzzy Rules/Hypercubes	5	3	4	4	3	3
No. of Parameters	55	33	30	32	24	27
RMSE of Training Process	0.057	0.007	0.016	0.0125	0.017	0.01
Training Time	10.35 s	15.71 s	203.49 s	15.55 s	11.88 s	13.95 s
RMSE of Testing Process	0.083	0.031	0.028	0.0288	0.034	0.025

Figure 8. The results of dynamic system identification with a longer input delay obtained using the STFCMAC model.

Figure 9. The errors of the system identification of longer input delays between the real output and the output obtained using the STFCMAC model.

5.3. Example 3: Prediction of Chaotic Time Series

The well-known Mackey–Glass chaotic time series prediction problem is used in this example. This chaotic time series is generated using the following delay differential equation:

$$\frac{du(t)}{dt} = \frac{0.2u(t-\tau)}{1+u^{10}(t-\tau)} - 0.1u(t) \quad (25)$$

The initial values are set as follows: $u(0) = 1.2$ and $\tau = 17$. In this study, four past values are used as inputs to predict $u(t)$. Therefore, the input–output data format is $[u(t-24), u(t-18), u(t-12), u(t-6), u(t)]$. Based on Equation (25), a total of 1000 data points are generated from $t = 124$ to $t = 1123$. The first 500 data points are used for training, whereas the remaining 500 are used for testing to validate the proposed model. The number of training epochs is set to 500. The initial parameters are set as follows: $\eta = 0.15$ and $\bar{S} = 0.0001$. After training, three fuzzy hypercube cells are generated. Table 3 compares the merits of various methods, including the rules, total numbers of parameters, and training and testing RMSEs. The performance of the STFCMAC model is compared with that of the D-FNN [34], G-FNN [27], TRFN-S [29], RSEFNN-LF [32], and PFCMAC [18] and that of neural learning models, namely, the SEELA [35], SuPFuNIS [36], and FWNN [37], as shown in Table 3. Figure 10; Figure 11 respectively illustrate the prediction results and errors between the actual output and output obtained using the STFCMAC model. The proposed STFCMAC model outperforms all of its competitors.

Table 3. A comparison of different predictors for chaotic time series prediction.

Models	FuzzyRules/Hypercubes	No. of Parameters	RMSE of Training Process	Training Time	RMSE of Testing Process
Mastorocostas and Theocharis [34]	10	100	-	-	0.0082
Gao and Er [27]	10	90	-	-	0.0056
Lin et al. [35]	9	198	0.0067	1375.38 s	0.0068
Paul and Kumar [36]	10	94	-	-	0.0057
Juang [29]	5	95	-	-	0.0124
Yilmaz and Oysal [37]	16	128	0.0023	27.80 s	0.0025
Juang et al. [32]	9	94	0.0032	17.44 s	0.0034
Lee et al. [18]	5	65	0.0028	731.17 s	0.0035
Proposed STFCMAC	3	42	0.0017	11.66 s	0.0022

Figure 10. The results of chaotic time series prediction obtained using the STFCMAC model.

Figure 11. The prediction errors of chaotic time series obtained using the STFCMAC model.

6. Conclusions

This study proposes a STFCMAC model with structure and parameter learning to solve identification and prediction problems, in which a simultaneous structure and parameter learning algorithm are proposed. Moreover, in the structure learning scheme, no initial structure exists in advance. That is, the proposed structure learning scheme can automatically determine the required structure of a network. Therefore, the proposed STFCMAC model has three advantages:

- (1) The proposed model requires less memory and fewer hypercubes/fuzzy rules.
- (2) The proposed model has a lower RMSE value.
- (3) The proposed model determines the number of hypercubes/fuzzy rules using the prespecified threshold value.

Inevitably, the proposed model has limitations. For example, determining a predetermined threshold value depends on user experience or trial and error. Therefore, an adaptive threshold selection in the STFCMAC model will be considered in future research. At the same time, in order to achieve high-speed operation in real-time applications, the STFCMAC model will be also implemented on a field programmable gate array in future research.

Author Contributions: Conceptualization, C.-J.L.; methodology, C.-J.L. and C.-H.L.; software, C.-H.L. and J.-Y.J.; data curation, C.-H.L. and J.-Y.J.; writing—original draft preparation, C.-J.L. and C.-H.L.; writing—review and editing, C.-J.L. and J.-Y.J.; funding acquisition, C.-J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Ministry of Science and Technology of the Republic of China, grant number MOST 107-2221-E-167-023.

Acknowledgments: The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan for financially supporting this research under Contract No. MOST 108-2221-E-167-026.

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

References

1. Abiodun, O.I.; Jantan, A.; Omolar, A.E.; Dada, K.V.; Mohamed, N.A.; Arshad, H. State-of-the-art in artificial neural network applications: A survey. *Heliyon* **2018**, *4*, e00938. [[CrossRef](#)] [[PubMed](#)]
2. Malekabadi, M.; Haghparast, M.; Nasiri, F. Air Condition's PID Controller Fine-Tuning Using Artificial Neural Networks and Genetic Algorithms. *Computers* **2018**, *7*, 32. [[CrossRef](#)]

3. Srivastava, S.; Sharma, L.; Sharma, V.; Kumar, A.; Darbari, H. Prediction of Diabetes Using Artificial Neural Network Approach: ICoEVCI 2018, India. In *Lecture Notes in Electrical Engineering*; Springer Science and Business Media LLC: New York, NY, USA, 2019. [\[CrossRef\]](#)
4. Caramazza, P.; Boccolini, A.; Buschek, D.; Hullin, M.; Higham, C.F.; Henderson, R.; Murray-Smith, R.; Faccio, D. Neural network identification of people hidden from view with a single-pixel, single-photon detector. *Sci. Rep.* **2018**, *8*, 11945. [\[CrossRef\]](#) [\[PubMed\]](#)
5. Albus, J.S. A new approach to manipulator control: The cerebellar model articulation controller. *Trans. ASME J. Dyn. Syst. Meas. Control* **1975**, *97*, 220–227. [\[CrossRef\]](#)
6. Albus, J.S. Data storage in the cerebellar model articulation controller. *Trans. ASME J. Dyn. Syst. Meas. Control* **1975**, *97*, 228–233. [\[CrossRef\]](#)
7. Ta, V.P.; Dang, X.K. An Innovative Recurrent Cerebellar Model Articulation Controller for Piezo-driven Micro-motion Stage. *Int. J. Innov. Comput. Inf. Control* **2018**, *14*, 1349–4198.
8. Zhou, X.; Li, Y.; Yue, H.; Jia, Y.; Zhao, L.; Zhu, Z. An improved cerebellar model articulation controller based on the compound algorithms of credit assignment and optimized smoothness for a three-axis inertially stabilized platform. *Mechatronics* **2018**, *53*, 95–108. [\[CrossRef\]](#)
9. Huang, M.L.; Lin, C.J. Nonlinear system control using a fuzzy cerebellar model articulation controller involving reinforcement-strategy-based bacterial foraging optimization. *Adv. Mech. Eng.* **2018**, *10*, 1–12. [\[CrossRef\]](#)
10. Almeida, P.E.M.; Simoes, M.G. Parametric CMAC networks: Fundamentals and applications of a fast convergence neural structure. *IEEE Trans. Ind. Appl.* **2003**, *39*, 1551–1557. [\[CrossRef\]](#)
11. Lin, C.M.; Peng, Y.F. Adaptive CMAC-based supervisory control for uncertain nonlinear systems. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 1248–1260. [\[CrossRef\]](#)
12. Shen, Z.; Guo, C.; Li, H. General fuzzified CMAC based model reference adaptive control for ship steering. *Proc. IEEE Int. Symp. Intell. Control* **2005**, *2005*, 1257–1262.
13. Sim, J.; Tung, W.L.; Quek, C. CMAC-Yager: A novel Yager-inference scheme-based fuzzy CMAC. *IEEE Trans. Neural Netw.* **2006**, *17*, 1394–1410. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Wu, T.F.; Tsai, P.S.; Chang, F.R.; Wang, L.S. Adaptive fuzzy CMAC control for a class of nonlinear systems with smooth compensation. *Proc. Inst. Electr. Eng. Control Theory Appl.* **2006**, *153*, 647–657. [\[CrossRef\]](#)
15. Lee, C.Y.; Lin, C.J.; Chen, H.J. A self-constructing fuzzy CMAC model and its applications. *Inf. Sci.* **2007**, *177*, 264–280. [\[CrossRef\]](#)
16. Macnab, C.J.B. Using RBFs in a CMAC to prevent parameter drift in adaptive control. *Neurocomputing* **2016**, *205*, 45–52. [\[CrossRef\]](#)
17. Zeng, X.J.; Keane, J.A. Approximation capabilities of hierarchical fuzzy systems. *IEEE Trans. Fuzzy Syst.* **2005**, *13*, 659–672. [\[CrossRef\]](#)
18. Lee, C.Y.; Lin, C.J.; Xu, Y.J. A parametric fuzzy CMAC model with hybrid evolutionary learning algorithms. *J. Mult.-Valued Log. Soft Comput.* **2007**, *13*, 89–114.
19. Chen, C.H. Design of TSK-type fuzzy controllers using differential evolution with adaptive mutation strategy for nonlinear system control. *Appl. Math. Comput.* **2013**, *219*, 8277–8294. [\[CrossRef\]](#)
20. Kuo, S.C.; Lee, C.L.; Lin, C.J. Applications of TAIEX and enrollment forecasting using an efficient improved fuzzy time series model. *Int. J. Innov. Comput. Inf. Control* **2016**, *12*, 459–466.
21. Chen, F.C.; Chang, C.H. Practical stability issues in CMAC neural network control systems. *IEEE Trans. Control Syst. Technol.* **1996**, *4*, 86–91. [\[CrossRef\]](#)
22. Hou, Z.G.; Gupta, M.M.; Nikiforuk, P.N.; Tan, M.; Cheng, L. A recurrent neural network for hierarchical control of interconnected dynamic systems. *IEEE Trans. Neural Netw.* **2007**, *18*, 466–481. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Maraziotis, I.A.; Dragomir, A.; Bezerianos, A. Gene networks reconstruction and time-series prediction from microarray data using recurrent neural fuzzy networks. *IET Syst. Biol.* **2007**, *1*, 41–50. [\[CrossRef\]](#) [\[PubMed\]](#)
24. Chen, C.S. TSK-type self-organizing recurrent-neural-fuzzy control of linear microstepping motor drives. *IEEE Trans. Power Electron.* **2010**, *25*, 2253–2265. [\[CrossRef\]](#)
25. Yang, S.C.; Lin, C.J.; Lin, H.Y.; Wang, J.G.; Yu, C.Y. Image Backlight Compensation Using Recurrent Functional Neural Fuzzy Networks Based on Modified Differential Evolution. *Iran. J. Fuzzy Syst.* **2016**, *13*, 1–19. [\[CrossRef\]](#)
26. Li, L.; Lin, C.J.; Huang, M.L.; Kuo, S.C.; Chen, Y.R. Mobile Robot Navigation Control Using Recurrent Fuzzy CMAC Based on Improved Dynamic Artificial Bee Colony. *Adv. Mech. Eng.* **2016**, *8*, 1–10. [\[CrossRef\]](#)

27. Gao, Y.; Er, M.J. NARMAX time series model prediction: Feedforward and recurrent fuzzy neural network approaches. *Fuzzy Sets Syst.* **2005**, *150*, 331–350. [[CrossRef](#)]
28. Theocharis, J.B. A high-order recurrent neuro-fuzzy system with internal dynamics: Application to the adaptive noise cancellation. *Fuzzy Sets Syst.* **2006**, *157*, 471–500. [[CrossRef](#)]
29. Juang, C.F. A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithm. *IEEE Trans. Fuzzy Syst.* **2002**, *10*, 155–170. [[CrossRef](#)]
30. Juang, C.F.; Lin, Y.Y.; Huang, R.B. Dynamic system modeling using a recurrent interval-valued fuzzy neural network and its hardware implementation. *Fuzzy Sets Syst.* **2011**, *179*, 83–99. [[CrossRef](#)]
31. Lin, Y.Y.; Chang, J.Y.; Lin, C.T. Identification and prediction of dynamic systems using an interactively recurrent self-evolving fuzzy neural network. *IEEE Trans. Neural Netw. Learn. Syst.* **2013**, *24*, 310–321. [[CrossRef](#)]
32. Juang, C.F.; Lin, Y.Y.; Tu, C.C. A recurrent self-evolving fuzzy neural network with local feedbacks and its application to dynamic system processing. *Fuzzy Sets Syst.* **2010**, *161*, 2552–2568. [[CrossRef](#)]
33. Lin, C.J.; Chin, C.C. Prediction and identification using wavelet-based recurrent fuzzy neural networks. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 2144–2154. [[CrossRef](#)] [[PubMed](#)]
34. Mastorocostas, P.A.; Theocharis, J.B. A recurrent fuzzy-neural model for dynamic system identification. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2002**, *32*, 176–190. [[CrossRef](#)] [[PubMed](#)]
35. Lin, C.J.; Chen, C.H.; Lin, C.T. Efficient self-evolving evolutionary learning for neuro-fuzzy inference systems. *IEEE Trans. Fuzzy Syst.* **2008**, *16*, 1476–1490.
36. Paul, S.; Kumar, S. Subsethood-product fuzzy neural inference system. *IEEE Trans. Neural Netw.* **2002**, *13*, 578–599. [[CrossRef](#)] [[PubMed](#)]
37. Yilmaz, S.; Oysal, Y. Fuzzy wavelet neural network models for prediction and identification of dynamic system. *IEEE Trans. Neural Netw.* **2010**, *21*, 1599–1609. [[CrossRef](#)]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).