Applied Soft Computing 18 (2014) 302–313

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A type-2 neural fuzzy system learned through type-1 fuzzy rules and its FPGA-based hardware implementation

Chia-Feng Juang*, Wen-Sheng Jang

Department of Electrical Engineering, National Chung Hsing University, Taichung 402, Taiwan, ROC

ARTICLE INFO

Article history:
Received 10 April 2013
Received in revised form 2 October 2013
Accepted 7 January 2014
Available online 23 January 2014

Keywords:
Type-2 fuzzy systems
Fuzzy neural networks
Neural fuzzy systems
Fuzzy chips
Fuzzy hardware

ABSTRACT

This paper first proposes a type-2 neural fuzzy system (NFS) learned through its type-1 counterpart (T2NFS-T1) and then implements the built IT2NFS-T1 in a field-programmable gate array (FPGA) chip. The antecedent part of each fuzzy rule in the T2NFS-T1 uses interval type-2 fuzzy sets, while the consequent part uses a Takagi-Sugeno-Kang (TSK) type with interval combination weights. The T2NFS-T1 uses a simplified type-reduction operation to reduce system training time and hardware implementation cost. Given a training data set, a TSK type-1 NFS is first learned through structure and parameter learning. The built type-1 fuzzy logic system (FLS) is then extended to a type-2 FLS, where highly overlapped type-1 fuzzy sets are merged into interval type-2 fuzzy sets to reduce the total number of fuzzy sets. Finally, the rule consequent and antecedent parameters in the T2NFS-T1 are tuned using a hybrid of the gradient descent and rule-ordered recursive least square (RLS) algorithms. Simulation results and comparisons with various type-1 and type-2 FLSs verify the effectiveness and efficiency of the T2NFS-T1 for system modeling and prediction problems. A new hardware circuit using both parallel-processing and pipeline techniques is proposed to implement the learned T2NFS-T1 in an FPGA chip. The T2NFS-T1 chip reduces the hardware implementation cost in comparison to other type-2 fuzzy chips.

1. Introduction

The neural-fuzzy approach to data-based modeling and prediction has drawn much research attention in the last two decades. While many neural fuzzy systems (NFSs) have been proposed, most studies have used type-1 fuzzy logic systems (FLSs) [1–4]. In recent years, studies on the theory and applications of interval type-2 FLSs have become a research focus. Interval type-2 FLSs are extensions of type-1 FLSs, where the membership value of an interval type-2 fuzzy set (FS) is an interval type-1 FS. Several advantages of using interval type-2 FLSs over their type-1 counterparts have been reported [5–7]. However, the footprint of uncertainty and operations with interval values in an interval type-2 FLS also lead to greater complexity in computing system outputs and assigning proper system parameters.

To automate the design of interval type-2 FLSs, several interval type-2 NFSs have been proposed with claimed superiority over the type-1 NFSs used for comparison [7–16]. Parameter learning of interval type-2 FLSs using a gradient descent algorithm was proposed in [7]. The approach does not use structure learning to determine the number of rules and FSs. In other words, the structure is fixed and should be assigned in advance. Several studies on structure learning of interval type-2 FLSs have been proposed [8–16]. These studies use the idea of clustering to generate type-2 fuzzy rules. The general approach is using the maximum value of centers of interval firing strengths as a rule generation criterion for each incoming datum [9–11,13–16]. Because structure learning is easier when using type-1 fuzzy rules than type-2 fuzzy rules, type-1 NFSs with structure learning ability have been extensively studied. This paper proposes a new method that builds a type-2 NFS via extending a built type-1 NFS (T2NFS-T1). For previous interval type-2 NFSs [8–15], the type-reduction outputs are computed using an iterative procedure, such as the Karnik–Mendel procedure [5], which is computationally expensive. For this problem, the T2NFS-T1 uses a simple weighted sum operation [17] to simplify the type-reduction operation, which reduces both software training time and hardware implementation cost. In learning, the T2NFS-T1 can be used to convert well-trained type-1 NFSs learned through different types of learning algorithms to type-2 NFSs without regeneration of the type-2 rules from an empty set. This is different from previous type-2 NFSs that generate type-2 fuzzy rules from an empty set [9–16], which do not make good use of the learning results in extensively studied type-1 NFSs. Given a training data set, a TSK type-1 NFS is first learned through structure and parameter learning. The T2NFS-T1 is then initialized from extending the built type-1 FLS to its type-2 counterpart. The general approach to the learning of an interval type-2 FLS from a type-1 FLS is by extending all type-1 FSs to interval type-2 FSs. In this

* Corresponding author. Tel.: +886 4 22840688x806; fax: +886 4 22851410.
E-mail address: cfjuang@dragon.nchu.edu.tw (C.-F. Juang).

1568-4946/$ – see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2014.01.006

direct conversion approach, the number of interval type-2 FSs is simply equal to the number of type-1 FSs in each variable. In the proposed extension approach, highly overlapped type-1 FSs are merged to interval type-2 FSs. This approach reduces the number of FSs in each input variable. That is, this operation improves FS transparency which is an important factor in the design of an interpretable FLS [18,19]. This learning approach is also different from most previous type-2 NFSs where the number of type-2 FSs is simply set be equal to the number of fuzzy rules [10,11,13,15]. Similarity measure of type-1 FSs is well-defined and is easier to compute than that of interval type-2 FSs [20]. As such, it is easier to merge type-1 FSs based on a similarity measure degree to reduce the total number of FSs in each input variable. The interval type-2 FLS adds extra degrees of freedom to a learned type-1 FLS, which would help to obtain a T2NFS-T1 that outperforms a type-1 NFS using the same rule base size. After initialization, parameter learning using a hybrid of the gradient descent and rule-ordered recursive least square (RLS) is used to tune the T2NFS-T1 to an optimal extent.

In addition to converting a type-1 FLS to a type-2 FLS using the T2NFS-T1, this paper proposes a new circuit to implement the learned type-2 FLS for the consideration of real applications. Several studies on the hardware implementation of interval type-2 FLSs have been proposed [10,16,21–27]. In contrast to type-1 fuzzy circuits [28–30], which process crisp values, interval type-2 fuzzy circuits process intervals, which require a relatively heavier computational load. In particular, finding the two interval boundary points in an interval type-2 FLS extended output requires an iterative procedure, such as the Karnik–Mendel procedure [5], and is costly in the designed chips [24–26]. To reduce the implementation cost, the circuit in [22] used Wu–Mendel closed form [31] for boundary point estimation. However, the computation load is still heavy for hardware implementation. A Look-Up-Table (LUT) circuit was proposed in Ref. [10,23] to store the left and right crossover points in advance for system output calculation, which avoids an iterative procedure. However, the LUT size grows exponentially with input dimensions and is most suitable to systems with low-dimensional inputs. In addition, this approach only applies to rules with constant consequent values. For the first-order Takagi-Sugeno-Kang (TSK)-type rules used in the hardware -implemented T2NFS-T1, the consequent values change with inputs, and therefore, the LUT circuit is not applicable. The simplified type-reduction operation [17] was used in an interval type-2 neural fuzzy chip with on-chip learning ability (IT2NFC-OL) to reduce the implementation cost [27]. This operation is also used in the hardware-implemented T2NFS-T1 (H-T2NFS-T1). In contrast to the IT2NFC-OL that uses Mamdani-type fuzzy rules, the H-T2NFS-T1 uses first-order TSK-type rules whose implementation using circuits is more complex and difficult. In addition, to accelerate the T2NFS-T1 chip execution speed, the paper uses four-pipeline in contrast to the two-pipeline technique in the IT2NFC-OL. The implementation of FLSs using personal computers is costly and inconvenient in some actual applications. In contrast to the personal computers, the H-T2NFS-T1 chip is much more suitable to actual applications that require small size and low-power consumptions in the implementation components, such as the fuzzy control problems [28,29,32]. In addition, the four-pipeline structure in the H-T2NFS-T1 chip is suitable to real-time operations that require high inference speed.

This paper is organized as follows. Section 2 introduces functions of the T2NFS-T1. Section 3 introduces the type-1 NFS based on which the T2NFS-T1 is built. Section 4 introduces the conversion of a type-1 FLS to a type-2 FLS and the subsequent parameter learning algorithms in the T2NFS-T1. Section 5 introduces FPGA implementation of a software-designed T2NFS-T1. Section 6 presents the software simulation implementation result of the T2NFS-T1. Section 7 presents the hardware implementation result. Finally, Section 8 presents the conclusion.

2. T2NFS-T1 functions

Each first-order TSK-type rule in the T2NFS-T1 has the following form:

$$\text{Rule } i: \text{ IF } x_1 \text{ is } \tilde{A}_1^i \quad \text{AND} \ldots \text{AND } x_n \text{ is } \tilde{A}_n^i \text{ THEN } y \text{ is } \tilde{a}_0^i + \sum_{j=1}^{n} \tilde{a}_j^i x_j, \quad i = 1, \ldots M, \tag{1}$$

where x_1, ..., x_n are input variables, y is output variable, $\tilde{A}_j^i$, $j = 1$, ..., n are interval type-2 FSs, M is the number of rules, and $\tilde{a}_j^i$'s, $j = 0$, ..., n are interval sets with the following representation:

$$\tilde{a}_j^i = [c_j^i - s_j^i, c_j^i + s_j^i], j = 0, \ldots, n. \tag{2}$$

The consequent value of each rule is an interval $[w_l^i, w_r^i]$, where the indices l and r represent the left and right limits, respectively. According to (1) and (2), the consequent interval value is

$$[w_l^i, w_r^i] = [c_0^i - s_0^i, c_0^i + s_0^i] + \sum_{j=1}^{n} [c_j^i - s_j^i, c_j^i + s_j^i] \cdot x_j. \tag{3}$$

That is,

$$w_l^i = \sum_{j=0}^{n} c_j^i x_j - \sum_{j=0}^{n} |x_j| s_j^i, w_r^i = \sum_{j=0}^{n} c_j^i x_j + \sum_{j=0}^{n} |x_j| s_j^i \tag{4}$$

where $x_0 := 1$. The output of the T2NFS-T1 is obtained via fuzzification, fuzzy meet, type-reduction, and defuzzification operations introduced as follows.

In the fuzzification operation, a Gaussian primary membership function having a fixed standard deviation σ_j^i and an uncertain mean that takes on values in $[m_{j1}^i, m_{j2}^i]$ is used, i.e.,

$$\mu_{\tilde{A}_j^i} = \exp\left\{-\frac{1}{2}\left(\frac{x_j - m_j^i}{\sigma_j^i}\right)^2\right\} \equiv N(m_j^i, \sigma_j^i; x_j), \quad m_j^i \in [m_{j1}^i, m_{j2}^i]. \tag{5}$$

The footprint of uncertainty of this membership function can be represented as a bounded interval in terms of an upper membership function, $\bar{\mu}_{\tilde{A}_j^i}$, and lower membership function, $\underline{\mu}_{\tilde{A}_j^i}$, where

$$\bar{\mu}_{\tilde{A}_j^i}(x_j) = \begin{cases} N(m_{j1}^i, \sigma_j^i; x_j) & x_j < m_{j1}^i \\ 1 & m_{j1}^i \le x_j \le m_{j2}^i \\ N(m_{j2}^i, \sigma_j^i; x_j) & x_j > m_{j2}^i \end{cases},$$

$$\underline{\mu}_{\tilde{A}_j^i}(x_j) = \begin{cases} N(m_{j2}^i, \sigma_j^i; x_j) & x_j \le \dfrac{m_{j1}^i + m_{j2}^i}{2} \\ N(m_{j1}^i, \sigma_j^i; x_j) & x_j > \dfrac{m_{j1}^i + m_{j2}^i}{2} \end{cases}. \tag{6}$$

That is, the output of the fuzzification operation can be represented as an interval type-1 fuzzy set $[\underline{\mu}_{\tilde{A}_j^i}, \bar{\mu}_{\tilde{A}_j^i}]$.

The fuzzy met operation finds the firing strength of each rule and is implemented using an algebraic product function [5]. The output is a firing strength, F^i, which is an interval type-1 FS given as follows:

$$F^i = [\underline{f}^i, \bar{f}^i] \tag{7}$$