# Fuzzy Regression Transfer Learning in Takagi-Sugeno Fuzzy Models

Hua Zuo, Guangquan Zhang, Witold Pedrycz, *Fellow, IEEE*, Vahid Bebood, Jie Lu, *Senior Member, IEEE*

***Abstract*—Data Science is a research field concerned with processes and systems that extract knowledge from massive amounts of data. In some situations, however, data shortage renders existing data-driven methods difficult or even impossible to apply. Transfer learning has recently emerged as a way of exploiting previously acquired knowledge to solve new yet similar problems much more quickly and effectively. In contrast to classical data-driven machine learning methods, transfer learning methods exploit the knowledge accumulated from data in auxiliary domains to facilitate predictive modeling in the current domain. A significant number of transfer learning methods that address classification tasks have been proposed, but studies on transfer learning in the case of regression problems are still scarce. This study focuses on using transfer learning techniques to handle regression problems in a domain that has insufficient training data. We propose an original fuzzy regression transfer learning method, based on fuzzy rules, to address the problem of estimating the value of the target for regression. A Takagi-Sugeno fuzzy regression model is developed to transfer knowledge from a source domain to a target domain. Experimental results using synthetic data and real world datasets demonstrate that the proposed fuzzy regression transfer learning method significantly improves the performance of existing models when tackling regression problems in the target domain.**

***Index Terms*—Fuzzy rules, fuzzy model, machine learning, regression, transfer learning.**

## I. Introduction

DATA science is an interdisciplinary field that involves the use of learning-based methods to analyze massive amounts of data and extract knowledge from them. Data-driven learning-based methods are not well-suited to situations where there is a data shortage, however, so the establishment of prediction models is impossible. This severely impedes the capacity of these methods to model in such environments.

What makes the human learning process advanced is our ability to transfer knowledge from one situation to another. Humans often draw upon more than just training for generalization: we continuously adapt to changing environments. Instead of learning from scratch, humans tackle new tasks on the basis of previously accumulated knowledge, and it is this unique ability that has inspired the application of transfer learning as a possible solution for the issue described above.

In recent years, research has increased the focus on transfer learning and its application to real world problems in the field of computational intelligence [1], [2]. Generally speaking, transfer learning addresses the problem of how to leverage previously acquired knowledge to improve the efficiency and accuracy of learning in another domain which in some way, or to some extent, relates to the original domain [3]. It is clear that transfer learning is needed to address real world problems. One well-known example of transfer learning is the indoor WiFi location estimation problem, which seeks to estimate a user's current location based on previously collected data [4], [5].

There have been many studies in the area of transfer learning, and related work can be divided into several categories, based on the problem setting: multi-task learning [6], domain adaptation [7], cross-domain adaptation [8], and heterogeneous learning [9]. When categorized by the approach applied, the main research streams include transferring knowledge of instances [10], [11], transferring knowledge of feature representations [12], transferring knowledge of model parameters [13], and transferring relational knowledge [14]. When classified from an application perspective, existing works can be generally categorized into three tasks: classification [15], [16], unsupervised learning (clustering [17], dimensionality deduction [18], [19]), and regression [20], [21]. There is a significant amount of research that studies transfer learning for classification problems, whereas studies on regression problems are still scarce.

In this paper, we focus on addressing regression problems using regression transfer learning techniques. Given that fuzzy system modeling is an important category of modeling with extensive applications [22], [23], incorporating regression transfer learning to a fuzzy model holds promise. Additionally, domains which lack information tend to suffer from uncertainty, and fuzzy regression transfer learning can cope efficiently with uncertainty. In transfer learning, target tasks in new environments often exhibit this uncertainty, especially when there is insufficient information, therefore a fuzzy system combined with transfer learning might exhibit a substantial

Manuscript received Feb. 29, 2016; accepted Sept. 29, 2016. This work was supported by the Australian Research Council under DP 140101366.

H. Zuo, G. Zhang, V. Behbood and J. Lu are with the Decision Systems & e-Service Intelligence Lab, Center for Quantum Computation & Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia (e-mail: Hua.Zuo@student.uts.edu.au; Guangquan.Zhang@uts.edu.au;Vahid.Behbood@uts.edu.au;Jie.Lu@uts.edu.au).

W. Pedrycz is with the Department of Electrical and Computer Engineering, University of Alberta, Canada (email: wpedrycz@ualberta.ca).

capacity to model uncertainty [24].

Several researchers have recently used fuzzy modeling mechanisms to solve transfer learning problems and improve the effectiveness of the model for the target domain [25] – [31]. The concept of information granularity has been introduced to knowledge transfer, providing a more general and advanced view of transfer learning problems [32]. These models regard parameters as information granules that assist with knowledge transfer, rather than numeric entities, and their specific form depends heavily on the problem [33].

The main contributions of this paper are twofold. First, a novel approach to regression, based on the Takagi-Sugeno fuzzy model, is introduced to address situations in which insufficient training data is available in the target domain but there is sufficient training data in the source domain. The Takagi-Sugeno model's fuzzy rules are constructed using source data, then modified through mappings to be reused to estimate values in the target domain. Second, this approach preserves the privacy of the source data because only the fuzzy rules are extracted.

The rest of this paper is structured as follows. The Takagi-Sugeno fuzzy model is introduced in Section II. Section III describes the new fuzzy regression transfer learning method based on fuzzy rules. The experiments and results used to analyze and verify the method are presented in Section IV. The final section concludes the paper and outlines future work.

## II. The Takagi-Sugeno Fuzzy Model

The proposed fuzzy regression transfer learning method transfers knowledge from a source domain to a target domain using a Takagi-Sugeno fuzzy model, which is an effective way to represent a fuzzy model in a nonlinear dynamic system. A Takagi-Sugeno fuzzy model composed of $c$ fuzzy rules is formally represented as:

Model $M$

$$\text{if } \boldsymbol{x} \text{ is } A_i(\boldsymbol{x}, \boldsymbol{v}_i), \text{ then } y \text{ is } L_i(\boldsymbol{x}, \boldsymbol{a}_i) \qquad i = 1,2, \dots, c \tag{1}$$

This fuzzy rule comprises one condition, which is described by the prototype $\boldsymbol{v}_i$, and one conclusion, which is typically governed by the linear function of the input variables.

The construction of this fuzzy rule-based model uses $\boldsymbol{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots, (\boldsymbol{x}_N, y_N)\}$ to formulate condition $A_i$ and optimize the parameters of the linear function $L_i$. The design procedure can be summarized as follows:

Step 1: Forming the conditions $A_1, A_2, \dots, A_c$ through fuzzy clustering

Typically, fuzzy c-means (FCM) is used to construct the clusters and calculate the prototypes $\boldsymbol{v}_i$. FCM partitions the $N$ data $\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_N$ into $c$ clusters, where $1 < c < N$. As a result, a collection of $c$ prototypes, $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_c$, and a partition matrix, $U = [u_{ik}], i = 1, 2, \dots, c, k = 1, 2, \dots, N$ are formed. The partition matrix satisfies two requirements $u_{ik} \in [0,1]$, $\sum_{i=1}^{c} u_{ik} = 1, \forall\, k$, and $0 < \sum_{k=1}^{N} u_{ik} < N, \forall\, i$.

The objective function (2) is minimized in the FCM:

$$J = \sum_{i=1}^{c} \sum_{k=1}^{N} (u_{ik})^m \|\boldsymbol{v}_i - \boldsymbol{x}_k\|^2 \tag{2}$$

where $\|\cdot\|$ stands for a distance function, and $m$ $(m > 1)$ is a fuzzification coefficient that affects the shape and overlap among the resulting membership functions.

Since real-world data has variables located in different ranges, a weighted Euclidean distance avoids bias towards any particular variable. The distance is expressed in the form

$$\|\boldsymbol{v}_i - \boldsymbol{x}_k\|^2 = \sum_{j=1}^{n} \frac{(v_{ij} - x_{kj})^2}{\sigma_j^2} \tag{3}$$

where $\sigma_j$ is the standard deviation of the $j$th feature, and $n$ is the dimensionality of the input data.

The prototypes are calculated as:

$$\boldsymbol{v}_i = \frac{\sum_{k=1}^{N} (u_{ik})^m \boldsymbol{x}_k}{\sum_{k=1}^{N} (u_{ik})^m} \tag{4}$$

and the entries of the partition matrix are expressed as follows:

$$u_{ik} = \frac{1}{\sum_{j=1}^{c} \left( \frac{\|v_i - x_k\|}{\|v_j - x_k\|} \right)^{\frac{2}{m-1}}} \tag{5}$$

The entire process is repeated until no significant changes to the entries of the partition matrix $U$ are reported in successive iterations of the algorithm.

Step 1 results in $c$ prototypes, $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_c$, that determine $A_1, A_2, \dots, A_c$, and $A_i(\boldsymbol{x}, \boldsymbol{v}_i)$ which can be calculated in the form:

$$A_i(\boldsymbol{x}, \boldsymbol{v}_i) = \frac{1}{\sum_{j=1}^{c} \left( \frac{\|x - v_i\|}{\|x - v_j\|} \right)^{\frac{2}{m-1}}} \tag{6}$$

Step 2: Optimizing the parameters of the linear function $L_i(\boldsymbol{x}, \boldsymbol{a}_i)$

Given a pair of data $(\boldsymbol{x}, y)$, where the input is $\boldsymbol{x}$, the output of the Takagi-Sugeno fuzzy model is denoted as $\hat{y}$ and determined as:

$$\hat{y} = \sum_{i=1}^{c} A_i(\boldsymbol{x}, \boldsymbol{v}_i)\, L_i(\boldsymbol{x}, \boldsymbol{a}_i) \tag{7}$$

where $L_i(\boldsymbol{x}, \boldsymbol{a}_i) = \boldsymbol{a}_i^T \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix}$, $\boldsymbol{a}_i$ is the coefficients vector of linear function $L_i$ in the form of $\boldsymbol{a}_i = [a_{i0}\ a_{i1}\ \cdots\ a_{in}]^T$, and $\boldsymbol{x}$ is n-dimensional input data, $\boldsymbol{x} = [x_1\ x_2\ \cdots\ x_n]^T$.

Next, we simplify (7):

$$\hat{y} = \sum_{i=1}^{c} A_i(\boldsymbol{x}, \boldsymbol{v}_i) \boldsymbol{a}_i^T \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix} = \sum_{i=1}^{c} \boldsymbol{a}_i^T \begin{bmatrix} A_i(\boldsymbol{x}, \boldsymbol{v}_i) \\ A_i(\boldsymbol{x}, \boldsymbol{v}_i)\boldsymbol{x} \end{bmatrix} \tag{8}$$

We denote $\boldsymbol{z}_i(\boldsymbol{x}) = \begin{bmatrix} A_i(\boldsymbol{x}, \boldsymbol{v}_i) \\ A_i(\boldsymbol{x}, \boldsymbol{v}_i)\boldsymbol{x} \end{bmatrix}$, then $\hat{y}$ can be rewritten as:

$$\hat{y} = \sum_{i=1}^{c} \boldsymbol{a}_i^T \boldsymbol{z}_i(\boldsymbol{x}) = \sum_{i=1}^{c} \boldsymbol{z}_i^T(\boldsymbol{x})\boldsymbol{a}_i = \boldsymbol{f}^T(\boldsymbol{x})\,\boldsymbol{a} \quad (9)$$

where $\boldsymbol{f}(\boldsymbol{x})^T = [\boldsymbol{z}_1(\boldsymbol{x})\ \boldsymbol{z}_2(\boldsymbol{x})\ \cdots\ \boldsymbol{z}_c(\boldsymbol{x})]$, $a = [\boldsymbol{a}_1\ \boldsymbol{a}_2 \cdots\ \boldsymbol{a}_c]^T$.

Therefore we find $\hat{y} = \boldsymbol{f}^T(\boldsymbol{x})\,\boldsymbol{a}$, i.e. for the given input $\boldsymbol{x}$, the output of the Takagi-Sugeno fuzzy model is parameter $\boldsymbol{a}$'s linear function. Additionally we expect that $\hat{y}$ will approximate $y$, which is the target value corresponding to $\boldsymbol{x}$. This can be achieved by an appropriate parameter $\boldsymbol{a}$ through a calculating process based on the dataset $\boldsymbol{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)\}$.

For all the inputs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$, the corresponding outputs are calculated as:

$$\hat{y}_i = \boldsymbol{f}^T(\boldsymbol{x}_i)\,\boldsymbol{a} \qquad i = 1,2,\cdots,N \quad (10)$$

We combine all the outputs and denote this as $\widehat{\boldsymbol{Y}}$:

$$\widehat{\boldsymbol{Y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \cdots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}^T(\boldsymbol{x}_1)\,\boldsymbol{a} \\ \boldsymbol{f}^T(\boldsymbol{x}_2)\,\boldsymbol{a} \\ \cdots \\ \boldsymbol{f}^T(\boldsymbol{x}_N)\,\boldsymbol{a} \end{bmatrix} = \boldsymbol{F}\boldsymbol{a} \quad (11)$$

where $\boldsymbol{F} = \begin{bmatrix} \boldsymbol{f}^T(\boldsymbol{x}_1) \\ \boldsymbol{f}^T(\boldsymbol{x}_2) \\ \cdots \\ \boldsymbol{f}^T(\boldsymbol{x}_N) \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_1^T(\boldsymbol{x}_1) & \boldsymbol{z}_2^T(\boldsymbol{x}_1) & \ldots & \boldsymbol{z}_c^T(\boldsymbol{x}_1) \\ \boldsymbol{z}_1^T(\boldsymbol{x}_2) & \boldsymbol{z}_2^T(\boldsymbol{x}_2) & \ldots & \boldsymbol{z}_c^T(\boldsymbol{x}_2) \\ \cdots & \cdots & \ldots & \cdots \\ \boldsymbol{z}_1^T(\boldsymbol{x}_N) & \boldsymbol{z}_2^T(\boldsymbol{x}_N) & \ldots & \boldsymbol{z}_c^T(\boldsymbol{x}_N) \end{bmatrix}$.

$\widehat{\boldsymbol{Y}}$ is expected to be as close as $\boldsymbol{Y} = [y_1 y_2\ \cdots\ y_N]^T$, which is the target value corresponding to inputs $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$.

Our aim is to minimize the objective function as follows:

$$Q = (\boldsymbol{F}\boldsymbol{a} - \boldsymbol{Y})^T(\boldsymbol{F}\boldsymbol{a} - \boldsymbol{Y}) \quad (12)$$

Since $Q$ is a quadratic function of $\boldsymbol{a}$, the optimal $\boldsymbol{a}$ can be obtained analytically

$$\boldsymbol{a}_{opt} = (\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T\boldsymbol{Y} \quad (13)$$

## III. Transfer Learning Based on the Takagi-Sugeno Fuzzy Model

The main idea of our fuzzy regression transfer learning method is presented in subsection A, and the procedures required to implement it are described in subsection B.

### *A. The Main Idea of the Fuzzy Regression Transfer Learning Method*

The Takagi-Sugeno fuzzy model $M$ for the source domain, constructed using pairs of data in the form of $(\boldsymbol{x}, y)$ and described in the form of the fuzzy rules, is:
Model $M$

if $\boldsymbol{x}$ is $A_i(\boldsymbol{x}, \boldsymbol{v}_i)$, then $y$ is $L_i(\boldsymbol{x}, \boldsymbol{a}_i)$ $\quad i = 1,2,\ldots,c$ (14)

When the input of fuzzy model $M$ is $\boldsymbol{x}$, the output of the model $y$ is an aggregate of the fuzzy rules as follows:

$$y = \sum_{i=1}^{c} A_i(\boldsymbol{x}, \boldsymbol{v}_i)\, L_i(\boldsymbol{x}, \boldsymbol{a}_i) \quad (15)$$

We also have new data from the target domain that are expressed in the form of the input-output pairs $(\boldsymbol{x}', g')$. However, the fuzzy model $M$ does not yield good performance with the data $(\boldsymbol{x}', g')$, since the new data follow the following fuzzy model $\widetilde{M}$ and the fuzzy rules, which are different to those in the fuzzy model $M$. To emphasize the changes in the parameters of the fuzzy model $M$, we describe it as:
Model $\widetilde{M}$

if $\boldsymbol{x}'$ is $A_i(\boldsymbol{x}', \boldsymbol{v}_i + \Delta\boldsymbol{v}_i)$, then $g'$ is $L_i(\boldsymbol{x}', \boldsymbol{a}_i + \Delta\boldsymbol{a}_i)$ (16)
$i = 1,2,\ldots,c$

The output $g'$ is therefore defined as (17) when the input is $\boldsymbol{x}'$:

$$g' = \sum_{i=1}^{c} A_i(\boldsymbol{x}', \boldsymbol{v}_i + \Delta\boldsymbol{v}_i)\, L_i(\boldsymbol{x}', \boldsymbol{a}_i + \Delta\boldsymbol{a}_i) \quad (17)$$

Given that there is insufficient data to train a new fuzzy model $\widetilde{M}$, we hope to use learned knowledge (fuzzy rules) in the existing model $M$ to help construct a fuzzy model that is more compatible with the new data. This requires the optimization of a continuous mapping of each input variable in the input space of the new data. The input space is transformed to $\Phi(\boldsymbol{x}')$ by mapping $\Phi$, and a new fuzzy model $M'$ is constructed using the fuzzy rules from fuzzy model $M$. The transformation process and resulting architecture are shown in Fig. 1.

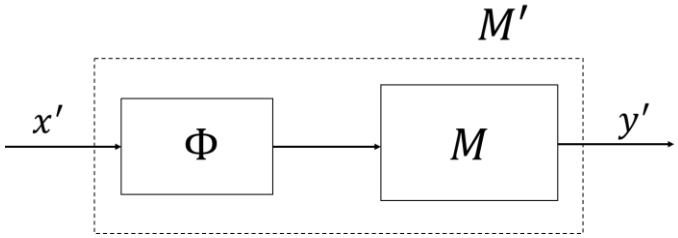



Fig. 1. Transfer knowledge in fuzzy rule-based models: a general concept.

Model $M'$, described in the form of fuzzy rules, is:
Model $M'$

if $\boldsymbol{x}'$ is $A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i))$, then $y'$ is $L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i)$ (18)
$i = 1,2,\ldots,c$

Therefore the output of model $M'$ is:

$$y' = \sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i))\ L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i) \quad (19)$$

Suppose the new dataset is $\boldsymbol{F} = \{(\boldsymbol{x}'_k, g'_k)\}$, and the number of data in $\boldsymbol{F}$ is $N'$, our aim is to find such $\Phi$ so that $M' \approx \widetilde{M}$, i.e.

$$\sum_{k=1}^{N'} y'_k \approx \sum_{k=1}^{N'} g'_k \quad (20)$$

or

$$\sum_{k=1}^{N'} \sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i))\, L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i) \approx \sum_{k=1}^{N'} g'_k \quad (21)$$

Mapping Φ is constructed by minimizing the objective function as follows:

$$Q' = \sum_{k=1}^{N'}(y'_k - g'_k) \tag{22}$$

The key to our fuzzy regression transfer learning method is to map the input space through Φ. A nonlinear continuous function based on sigmoid functions is used to construct the mapping Φ. Other forms of mapping suitable for the problem could also be considered.

Fig. 2 shows the structure of nonlinear continuous mapping for each input variable using the $i$th input variable of data $\boldsymbol{x}'_k$ as an example.

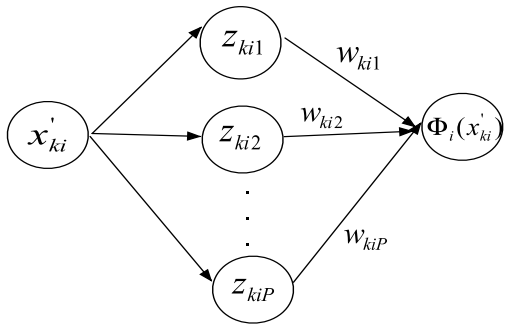



Fig. 2. Architecture of nonlinear mapping

The mapping for each input variable is constructed through a network that is composed of $P$ nodes situated in the hidden layer and a single node placed at the output layer. The nodes in the hidden layer are two-parametric sigmoid functions, where the $p$th sigmoid function for the $i$th input variable of $\boldsymbol{x}'_k$ is:

$$z_{kip} = \frac{1}{1+e^{-\alpha_{ip}(x'_{ki}-\beta_{ip})}} \tag{23}$$
$$i = 1,2,\dots,n,\ p = 1,2,\dots,P,\ \alpha_{ip} > 0$$

The transformation of input variable $x'_{ki}$ under mapping $\Phi_i$ is therefore:

$$\Phi_i(x'_{ki}) = \sum_{p=1}^{P} w_{kip} z_{kip} \tag{24}$$

where $w_{kip}$ represents the weight of the $p$th sigmoid function to the output, and satisfies $\sum_{p=1}^{P} w_{kip} = \max x'_i = \max\{x'_{1i}, x'_{2i}, \cdots, x'_{N'i}\}$. $z_{kip}$ is calculated by (23).

By taking advantage of the nonlinear mappings, the input space is transformed so that the new input variables become more compatible with the fuzzy rules of the existing fuzzy model.

Estimating an "optimal" number of clusters in the clustering algorithms is also still an open issue, and the solution to this problem greatly depends upon the data and the problem. In this paper, we adopt a strategy of dynamically changing the number of clusters (prototypes). The number of clusters (C) is chosen from a certain range and a fuzzy regression transfer learning model is constructed for each selected C. The model with the best transfer performance is then chosen to address the problem at hand.

### *B. Transfer Knowledge in Fuzzy Rule-based Models*

The procedure for transferring knowledge from a source domain to a target domain requires two steps. First a Takagi-Sugeno fuzzy model, based on source data, is constructed, then a new fuzzy model for the target domain is built by modifying the input space using fuzzy rules from the source domain.

Step 1: Constructing a Takagi-Sugeno fuzzy model $M$ based on source data

Suppose the dataset in the source domain is $\boldsymbol{D} = \{(\boldsymbol{x}_k, y_k)\}$, and the number of data in $\boldsymbol{D}$ is $N$. Based on the dataset $\boldsymbol{D}$, a Takagi-Sugeno fuzzy model $M$ for the source domain is constructed.

Model $M$

$$\text{if } \boldsymbol{x}_k \text{ is } A_i(\boldsymbol{x}_k, \boldsymbol{v}_i), \text{ then } y_k \text{ is } L_i(\boldsymbol{x}_k, \boldsymbol{a}_i) \tag{25}$$
$$i = 1,2,\dots,c.$$

The main blocks of a fuzzy rule are the condition and conclusion, which are dominated by the prototype and linear function respectively. The fuzzy model $M$ is constructed by calculating the prototypes of the data and estimating the parameters of the linear functions standing in the conclusions of the rules.

*1) Forming the prototypes*

Fuzzy c-means is used to cluster the input data $\{\boldsymbol{x}_k\}$ and find the prototypes:

$$\boldsymbol{V} = [\boldsymbol{v}_1\ \boldsymbol{v}_2\ \cdots\ \boldsymbol{v}_c]^T \tag{26}$$

where $c$ is the number of clusters, i.e. the number of fuzzy rules.

The input $\boldsymbol{x}_k$ therefore belongs to the prototype (cluster) $\boldsymbol{v}_i$ with the membership degree $A_i(\boldsymbol{x}_k, \boldsymbol{v}_i)$, which is calculated as follows:

$$A_i(\boldsymbol{x}_k, \boldsymbol{v}_i) = \frac{1}{\sum_{j=1}^{c}\left(\frac{\|v_i - x_k\|}{\|v_j - x_k\|}\right)^{\frac{2}{m-1}}} \tag{27}$$

*2) Developing linear functions*

Since the functions in the conclusion are linear, they are uniquely described by the coefficients $\boldsymbol{a}_i$, $i = 1,2,\dots,c$. Based on the analysis completed in Section II, the coefficients of the linear function are calculated as follows:

$$[\boldsymbol{a}_1\ \boldsymbol{a}_2\ \cdots\ \boldsymbol{a}_c] = (\boldsymbol{F}^T\boldsymbol{F})^{-1}\boldsymbol{F}^T\boldsymbol{Y} \tag{28}$$

where $\boldsymbol{F} = [\boldsymbol{f}^T(\boldsymbol{x}_1)\ \boldsymbol{f}^T(\boldsymbol{x}_2)\ \cdots\ \boldsymbol{f}^T(\boldsymbol{x}_N)]^T$, $\boldsymbol{f}^T(\boldsymbol{x}_k) = \begin{bmatrix} A_1(\boldsymbol{x}_k, \boldsymbol{v}_1) & \dots & A_c(\boldsymbol{x}_k, \boldsymbol{v}_c) \\ A_1(\boldsymbol{x}_k, \boldsymbol{v}_1)\boldsymbol{x}_k & \dots & A_c(\boldsymbol{x}_k, \boldsymbol{v}_c)\boldsymbol{x}_k \end{bmatrix}$

$[\boldsymbol{a}_1\ \boldsymbol{a}_2\ \cdots\ \boldsymbol{a}_c]$ is the coefficient matrix of the linear functions, where $\boldsymbol{a}_i = [a_{i0}\ a_{i1}\ \cdots\ a_{in}]^T$ is the coefficient vector of the $i$th linear function $L_i$, $i = 1,2,\dots,c$.

As a consequence, the prototypes and linear functions are calculated to construct the fuzzy rules in (25). When a new datum $\boldsymbol{x}$ appears, the output of fuzzy model $M$ is calculated as follows:

$$y = \sum_{i=1}^{c} A_i(\boldsymbol{x}, \boldsymbol{v}_i) L_i(\boldsymbol{x}, \boldsymbol{a}_i) \tag{29}$$

where $A_i(\boldsymbol{x}, \boldsymbol{v}_i) = 1/\sum_{j=1}^{c} \left(\frac{\|x - v_i\|}{\|x - v_j\|}\right)^{\frac{2}{m-1}}$, $L_i(\boldsymbol{x}, \boldsymbol{a}_i) = a_{i0} + a_{i1}x_1 + \cdots + a_{in}x_n$.

This fuzzy model $M$ will not perform well on the target data $\boldsymbol{F} = \{(\boldsymbol{x}'_k, g'_k)\}$, $k = 1,2,\dots,N'$, $N' \ll N$, however. To improve its performance, the input space of the target domain must be made more compatible with $M$'s fuzzy rules.

Step 2: Modifying the input space and using the fuzzy rules of the existing model

The input space is modified by constructing a mapping for each input variable, say $\Phi_i(x'_{ki})$, where $x'_{ki}$ is the $i$th input variable of $\boldsymbol{x}'_k$. The input space is thus transformed by $\Phi = [\Phi_1\ \Phi_2\ \cdots\ \Phi_n]$, and the input data $\boldsymbol{x}'_k$ becomes $\Phi(\boldsymbol{x}'_k)$. The specific form of $\Phi(\boldsymbol{x}'_k)$ depends on the construction method for $\Phi$. As described in Part *A*, nonlinear continuous mappings, shown in Fig. 2, are used here, and the corresponding $\Phi(\boldsymbol{x}'_k)$ is calculated through (30):

$$\Phi(\boldsymbol{x}'_k) = \begin{bmatrix} \Phi_1(x'_{k1}) \\ \Phi_2(x'_{k2}) \\ \cdots \\ \Phi_n(x'_{kn}) \end{bmatrix} = \begin{bmatrix} \sum_{p=1}^{P} w_{1p}/\left(1 + e^{-\alpha_{1p}(x'_{k1} - \beta_{1p})}\right) \\ \sum_{p=1}^{P} w_{2p}/\left(1 + e^{-\alpha_{2p}(x'_{k2} - \beta_{2p})}\right) \\ \cdots \\ \sum_{p=1}^{P} w_{np}/\left(1 + e^{-\alpha_{np}(x'_{kn} - \beta_{np})}\right) \end{bmatrix} \tag{30}$$

Since $\Phi$ transforms the whole input space, both the dataset $\{\boldsymbol{x}'_k\}$ and the prototypes in (26) need to be transformed using the mapping $\Phi$. The prototype $\boldsymbol{v}_i$ becomes $\Phi(\boldsymbol{v}_i)$. Model $M$'s fuzzy rules are transferred to the new data space, and a new model $M'$ is built.

Model $M'$

$$\text{if } \boldsymbol{x}'_k \text{ is } A_i(\Phi(\boldsymbol{x}'_k), \Phi(\boldsymbol{v}_i)), \text{ then } y' \text{ is } L_i(\Phi(\boldsymbol{x}'_k), \boldsymbol{a}_i) \tag{31}$$

where $A_i(\Phi(\boldsymbol{x}'_k), \Phi(\boldsymbol{v}_i)) = 1/\sum_{j=1}^{c} \left(\frac{\|\Phi(x'_k) - \Phi(v_i)\|}{\|\Phi(x'_k) - \Phi(v_j)\|}\right)^{\frac{2}{m-1}}$,

$L_i(\Phi(\boldsymbol{x}'_k), \boldsymbol{a}_i) = a_{i0} + a_{i1}\Phi_1(x'_{k1}) + \cdots + a_{in}\Phi_n(x'_{kn})$, $i = 1,2,\dots,c$.

Therefore, when the input is $\boldsymbol{x}'_k$, the output of model $M'$ is:

$$y'_k = \sum_{i=1}^{c} A_i\big(\Phi(\boldsymbol{x}'_k), \Phi(\boldsymbol{v}_i)\big)\ L_i(\Phi(\boldsymbol{x}'_k), \boldsymbol{a}_i) \tag{32}$$

For all the input data in dataset $\{\boldsymbol{x}'_k\}$, we find the output $\{y'_k\}$ corresponding to (32).

The parameters of $\Phi$ are optimized by minimizing (33). In the literature, Particle Swarm Optimization (PSO) and Differential Evolution (DE) are reported to be two of the most suitable global optimization algorithms [34]. In this paper, we apply PSO and DE to optimize the parameters, and their performance in the fuzzy regression transfer learning is compared in detail in Section IV.

$$Q' = \frac{1}{N'}\sum_{k=1}^{N'}(y'_k - g'_k)^2 \tag{33}$$

## IV. Experimental Studies

To evaluate the proposed fuzzy regression transfer learning method and its learning algorithm, both synthetic and real-world datasets are adopted, as will be described in Sections IV-A and IV-B. Section IV-A validates the new method and explores the impact of optimization algorithms in the new method. Section IV-B elaborates on the usefulness of the new fuzzy regression transfer learning technology in dealing with real world problems.

### *A. Synthetic Data*

The focus of this paper is on using the new fuzzy regression transfer technology to address regression problems. Data outputs in regression problems are more complicated than outputs in classification problems, and as a result of the fuzzy model we use, the method of generating the synthetic data is crucial and must be reasonable. Therefore, prior to demonstrating the results of the experiments, we will describe the process of generating the source data and target data before constructing the fuzzy models based on the formula presented in Section III. A number of symbolic representations of the experimental results are denoted in this section so that the presentation of the experiment results is clearer. Section IV-A-1 presents the data generation and model construction process, and Section IV-A-2 outlines the experimental results.

#### *1) Data generation and model construction*

The procedures for the experiments in this section are more detailed and integrated than those in Section III-B, as they include an additional step, Step 3, to verify that the model using our method performs better than the model trained using scarce target data. The steps of the procedure are as follows:

Step 1: Generate source data and construct fuzzy model $M$.

Step 2: Generate target data, the number of which is much smaller than the number in the source domain, and use the existing model $M$ to estimate the outputs of the target data.

Step 3: Use the data in the target domain to construct a new model $\bar{M}$ for the target domain.

Step 4: Modify the input space of model $M$ using the target data to obtain a new model $M'$ for the target domain.

We use the 5-fold cross validation, which is commonly used in model validation in machine learning, to construct the models in Steps 1, 3 and 4.

More detail follows about these four steps, especially concerning the generation of the datasets.

Step 1: Generate source data and construct the fuzzy model $M$.

This step is divided into two sub-steps.

Step1-1: Generate source data.

The process of generating source data includes the generation of the inputs and finding the corresponding outputs to constitute the input-output pairs. Referring to the input data $\boldsymbol{X}$, we generate three sub-datasets $\boldsymbol{X}_1, \boldsymbol{X}_2, \boldsymbol{X}_3$ that have different distributions, and combine them to construct $\boldsymbol{X}$. The parameters related to the input data are listed in Table I.

TABLE I
PARAMETERS OF INPUT DATA IN THE SOURCE DOMAIN

| |
|---|
| $\boldsymbol{X}_1 = \{\boldsymbol{x}_k\}, k = 1, \dots, N/3, \boldsymbol{x}_k \sim N(\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1)$ |
| $\boldsymbol{X}_2 = \{\boldsymbol{x}_k\}, k = N/3 + 1, \dots, 2N/3, \boldsymbol{x}_k \sim N(\boldsymbol{\mu}_2, \boldsymbol{\sigma}_2)$ |
| $\boldsymbol{X}_3 = \{\boldsymbol{x}_k\}, k = 2N/3 + 1, \dots, N, \boldsymbol{x}_k \sim N(\boldsymbol{\mu}_3, \boldsymbol{\sigma}_3)$ |

We generate the input data $\boldsymbol{X} = \{\boldsymbol{x}_k\}, k = 1, \dots, N$, where $N$ is the number of source data in the source domain. Based on this, we calculate the output data in the following way. For the sake of reasonability, the outputs are generated according to (29), so the prototypes and linear function are needed in advance. Since the datasets follow normal distributions, we assume the mean values of the normal distributions as the prototypes, denoted as $\overline{\boldsymbol{v}}$:

$$\overline{\boldsymbol{v}} = [\overline{\boldsymbol{v}}_1\ \overline{\boldsymbol{v}}_2\ \overline{\boldsymbol{v}}_3]^T = [\boldsymbol{\mu}_1\ \boldsymbol{\mu}_2\ \boldsymbol{\mu}_3]^T \tag{34}$$

The coefficients of linear functions $\overline{\boldsymbol{a}}_i$, $i = 1,2,3$, are given in Table II.

TABLE II
COEFFICIENTS OF LINEAR FUNCTIONS IN THE SOURCE DOMAIN

| | |
|---|---|
| $L_1(\cdot, \overline{\boldsymbol{a}}_1)$ | $\overline{\boldsymbol{a}}_1 = [\bar{a}_{10}\ \ \bar{a}_{11}\ \ \bar{a}_{12}]$ |
| $L_2(\cdot, \overline{\boldsymbol{a}}_2)$ | $\overline{\boldsymbol{a}}_2 = [\bar{a}_{20}\ \ \bar{a}_{21}\ \ \bar{a}_{22}]$ |
| $L_3(\cdot, \overline{\boldsymbol{a}}_3)$ | $\overline{\boldsymbol{a}}_3 = [\bar{a}_{30}\ \ \bar{a}_{31}\ \ \bar{a}_{32}]$ |

Based on the prototypes and linear functions, when the input is $\boldsymbol{x}_k$, the output $y_k$ can be obtained as follows:

$$y_k = \sum_{i=1}^{c} A_i(\boldsymbol{x}, \overline{\boldsymbol{v}}_i) L_i(\boldsymbol{x}, \overline{\boldsymbol{a}}_i) \tag{35}$$

We calculate the output $\boldsymbol{Y} = \{y_k\}, k = 1, \dots, N$, and finally obtain the data $\boldsymbol{D} = (\boldsymbol{X}, \boldsymbol{Y})$.

Step 1-2: Construct the fuzzy model $M$ based on data $(\boldsymbol{X}, \boldsymbol{Y})$.

The construction of a Takagi-Sugeno fuzzy model depends on the prototypes $\boldsymbol{v}_i$ and linear functions $L_i(\cdot, \boldsymbol{a}_i)$. The details are discussed in Step 1 in Section III-B.

We apply the 5-fold cross validation when constructing model $M$, and split the dataset $\boldsymbol{D}$ into a training set $\boldsymbol{D}_1$(80%) and a testing set $\boldsymbol{D}_2$ (20%). Once model $M$ has been constructed, it is tested on the testing set $\boldsymbol{D}_2$, and the mean square error (MSE) is denoted as:

$$Q = \frac{1}{N_2} \sum_{k=1}^{N_2} (y_k - d_k)^2 \tag{36}$$

where $d_k$ is the output of model $M$ when its input is $\boldsymbol{x}_k$, $(\boldsymbol{x}_k, y_k) \in \boldsymbol{D}_2$. $N_2$ is the number of data in the testing set $\boldsymbol{D}_2$. We calculate the mean and standard deviation of $Q$.

Step 2: Generate target data, the number of which is much smaller than the number in the source domain, and use the existing model $M$ to estimate the outputs of the target data.

Step 2 is also divided into two sub-steps to better describe it.

Step 2-1: Generate target data.

The method of generating input data $\boldsymbol{X}'$ in the target domain is the same as the method used in the source domain, and the parameters related to the input data in the target domain are listed in Table III.

TABLE III
PARAMETERS OF INPUT DATA IN THE TARGET DOMAIN

| |
|---|
| $\boldsymbol{X}'_1 = \{\boldsymbol{x}'_k\}, k = 1, \dots, N'/3, \boldsymbol{x}'_k \sim N(\boldsymbol{\mu}'_1, \boldsymbol{\sigma}'_1)$ |
| $\boldsymbol{X}'_2 = \{\boldsymbol{x}'_k\}, k = N'/3 + 1, \dots, 2N'/3, \boldsymbol{x}'_k \sim N(\boldsymbol{\mu}'_2, \boldsymbol{\sigma}'_2)$ |
| $\boldsymbol{X}'_3 = \{\boldsymbol{x}'_k\}, k = 2N'/3 + 1, \dots, N', \boldsymbol{x}'_k \sim N(\boldsymbol{\mu}'_3, \boldsymbol{\sigma}'_3)$ |

The prototypes of the clusters are also the mean values:

$$\overline{\boldsymbol{v}}' = [\overline{\boldsymbol{v}}'_1\ \overline{\boldsymbol{v}}'_2\ \overline{\boldsymbol{v}}'_3]^T = [\boldsymbol{\mu}'_1\ \boldsymbol{\mu}'_2\ \boldsymbol{\mu}'_3]^T \tag{37}$$

The coefficients of the linear functions in the target domains are given in Table IV.

TABLE IV
COEFFICIENTS OF LINEAR FUNCTIONS IN THE TARGET DOMAIN

| | |
|---|---|
| $L_1(\cdot, \overline{\boldsymbol{a}}'_1)$ | $\overline{\boldsymbol{a}}'_1 = [\bar{a}'_{10}\ \ \bar{a}'_{11}\ \ \bar{a}'_{12}]$ |
| $L_2(\cdot, \overline{\boldsymbol{a}}'_2)$ | $\overline{\boldsymbol{a}}'_2 = [\bar{a}'_{20}\ \ \bar{a}'_{21}\ \ \bar{a}'_{22}]$ |
| $L_3(\cdot, \overline{\boldsymbol{a}}'_3)$ | $\overline{\boldsymbol{a}}'_3 = [\bar{a}'_{30}\ \ \bar{a}'_{31}\ \ \bar{a}'_{32}]$ |

To make the source and target data different, the prototypes and linear functions in the source domain and target domain must also be different, i.e.

$$\overline{\boldsymbol{v}}'_i \neq \overline{\boldsymbol{v}}_i, \quad \overline{\boldsymbol{a}}'_i \neq \overline{\boldsymbol{a}}_i \tag{38}$$

where $\boldsymbol{v}_i$, $\overline{\boldsymbol{a}}_i$ and $\overline{\boldsymbol{v}}'_i$, $\overline{\boldsymbol{a}}'_i$ are the prototypes and coefficients of the linear functions in the source domain and target domain, respectively.

Except for the prototypes, the covariance matrixes of the target data are not the same as those of the source data. This variety in the target data is beneficial for testing the validity of our algorithm.

Based upon the input data $\boldsymbol{X}'$, the corresponding output $\boldsymbol{G}' = \{g'_k\}$ is calculated as follows:

$$g'_k = \sum_{i=1}^{c} A_i(\boldsymbol{x}'_k, \boldsymbol{v}'_i) L_i(\boldsymbol{x}'_k, \boldsymbol{a}'_i) \tag{39}$$

As a consequence, we have the data $\boldsymbol{F} = (\boldsymbol{X}', \boldsymbol{G}') = \{(\boldsymbol{x}'_k, g'_k)\}$ in the target domain. The number of data in $\boldsymbol{F}$ is much smaller than in $\boldsymbol{D}$ ($N' \ll N$).

Step 2-2: Use the existing model $M$ to estimate the outputs of the target data.

Given that the source domain and target domain have different prototypes and linear functions, the fuzzy model $M$ does not perform well on the target data. This means that when the input is $\boldsymbol{x}'_k$, the output $h'_k$ calculated using the fuzzy rules for the source data may be quite different with $g'_k$. We test model $M$ on target data $\boldsymbol{F}$, and the discrepancy between $g'_k$ and $h'_k$ is denoted as $Q_1$:

$$Q_1 = \frac{1}{N'}\sum_{k=1}^{N'}(h'_k - g'_k)^2 \tag{40}$$

where $h'_k$ is the output of model $M$ when the input is $\boldsymbol{x}'_k$, i.e. $h'_k = \sum_{i=1}^{c} A_i(\boldsymbol{x}'_k, \boldsymbol{v}_i)L_i(\boldsymbol{x}'_k, \boldsymbol{a}_i)$ , $(\boldsymbol{x}'_k,\ g'_k) \in \boldsymbol{F}$ , $N'$ is the number of data in the target domain.

Step 3: Use the data in the target domain to construct a new model $\bar{M}$ for the target domain.

Proving that a model does not perform as well when trained with less data in the target domain supports our assumption. Although only a small amount of data is available in the target domain, they can nevertheless be used to train a model; the problem is that the accuracy of the model cannot be guaranteed since the training data is insufficient. The procedures for the construction of $\bar{M}$ are exactly the same as are used to build model $M$ for the source domain, and we also apply 5-fold cross validation procedure. The target dataset $\boldsymbol{F}$ is split into a training set $\boldsymbol{F}_1$(80%) and a testing set $\boldsymbol{F}_2$(20%). Model $\bar{M}$ is constructed based on the training set $\boldsymbol{F}_1$ and then used to predict the outputs for the testing set $\boldsymbol{F}_2$. The MSE is denoted as:

$$Q_2 = \frac{1}{N'_2}\sum_{k=1}^{N'_2}(s'_k - g'_k)^2 \tag{41}$$

where $s'_k$ is the output of model $\bar{M}$ when its input is $\boldsymbol{x}'_k$, $(\boldsymbol{x}'_k,\ g'_k) \in \boldsymbol{F}_2$, $N'_2$ is the number of data in the testing set $\boldsymbol{F}_2$. The mean and standard deviation of $Q_2$ are obtained.

Step 4: Modify the input space of model $M$ using target data to obtain a new model $M'$ for the target domain.

The input space of model $M$ is modified by the continuous mapping $\Phi$, which is obtained by minimizing (42) using the training set $\boldsymbol{F}_1$.

$$Q' = \frac{1}{N'_1}\sum_{k=1}^{N'_1}(y'_k - g'_k)^2 \tag{42}$$

where $y'_k$ is the output of model $M'$ calculated by (32) when the input is $\boldsymbol{x}'_k$, $(\boldsymbol{x}'_k,\ g'_k) \in \boldsymbol{F}_1$, $N'_1$ is the number of data in the training set $\boldsymbol{F}_1$. The optimization algorithms PSO and DE are used to find the optimal parameters of $\Phi$.

Following the construction of model $M'$, it is tested on the testing set $\boldsymbol{F}_2$, and the MSE is denoted as:

$$Q_3 = \frac{1}{N'_2}\sum_{k=1}^{N'_2}(t'_k - g'_k)^2 \tag{43}$$

where $t'_k$ is the output of model $M'$ when the input is $\boldsymbol{x}'_k$, $(\boldsymbol{x}'_k,\ g'_k) \in \boldsymbol{F}_2$, $N'_2$ is the number of data in the testing set $\boldsymbol{F}_2$.

The above process also involves a 5-fold cross validation procedure, so the mean and standard deviation of $Q_3$ are calculated.

In the following experiments, $Q_1, Q_2$ and $Q_3$ are compared, and the desired outcome is that $Q_3$ will be smaller than both $Q_1$ and $Q_2$ . $Q_3 < Q_1$ shows that the modified model $M'$ is improved compared to the existing model $M$, and $Q_3 < Q_2$ demonstrates that the modified model $M'$ is better than model $\bar{M}$ trained using few data in the target domain. Next, the experimental results are given to verify the effectiveness of the proposed fuzzy regression transfer learning method.

*2) Experiments*

The details of three experiments are provided in this section. The first experiment illustrates the application scope of our method. The second compares the outcomes of using PSO and DE, and validates the new fuzzy regression transfer learning method. The third experiment discusses an alternative method of constructing the mapping for the input space, and compares it with the method shown in Fig. 2 when the interactions between the features are considered.

Experiment 1

The assumption is that the data in the target domain is insufficient to train a good model; however, because of the characteristics of the model we use, i.e. a Takagi-Sugeno fuzzy model, there is one special situation in which, even though the number of available data in the target domain is small, the data will construct a model that performs well.

The fuzzy rules of the Takagi-Sugeno fuzzy model are distinguished by the data partitions, so when the data has good division of clusters, even if the number of data is small, a good model can be built without the need to transfer knowledge from another domain, which may even be harmful. However, this situation is dependent on the distribution of the data, and is rare in real applications, so our method is able to handle most scenarios in transfer learning. In Experiment 1, we provide an example of this special situation where, although the data in the target domain is scarce, there is no need to transfer knowledge from other domains.

Source data and target data both follow normal distribution. The mean values and covariance matrixes of the source data and target data in Experiment 1 are shown in Table V.

TABLE V
DISTRIBUTIONS OF SOURCE DATA AND TARGET DATA

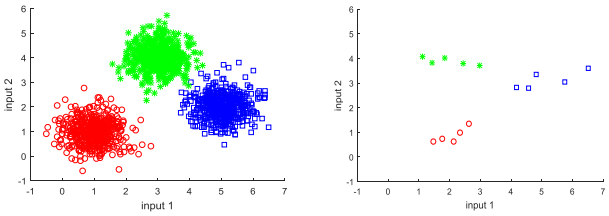
| Source data | | Target data | |
|---|---|---|---|
| Mean values | Covariance | Mean values | Covariance |
| $\boldsymbol{\mu}_1 = [1 \quad 1]$ | $\boldsymbol{\sigma}_1 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$ | $\boldsymbol{\mu}'_1 = [2 \quad 1]$ | $\boldsymbol{\sigma}'_1 = \begin{bmatrix} 0.4^2 & 0.1 \\ 0.1 & 0.4^2 \end{bmatrix}$ |
| $\boldsymbol{\mu}_2 = [5 \quad 2]$ | $\boldsymbol{\sigma}_2 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$ | $\boldsymbol{\mu}'_2 = [5 \quad 3]$ | $\boldsymbol{\sigma}'_2 = \begin{bmatrix} 0.5^2 & 0.1 \\ 0.1 & 0.5^2 \end{bmatrix}$ |
| $\boldsymbol{\mu}_3 = [3 \quad 4]$ | $\boldsymbol{\sigma}_3 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$ | $\boldsymbol{\mu}'_3 = [2 \quad 4]$ | $\boldsymbol{\sigma}'_3 = \begin{bmatrix} 0.5^2 & 0.1 \\ 0.1 & 0.5^2 \end{bmatrix}$ |

The linear functions in the source domain and target domain shown in Table VI are also different.

TABLE VI
THE COEFFICIENTS OF LINEAR FUNCTIONS IN TWO DOMAIN

| Source domain | | Target domain | |
|---|---|---|---|
| $L_1(\bar{a}_1)$ | $\bar{a}_1 = [1 \quad 1 \quad 1]$ | $L_1(\bar{a}_1')$ | $\bar{a}_1' = [1.5 \quad 0.5 \quad 1.5]$ |
| $L_2(\bar{a}_2)$ | $\bar{a}_2 = [2 \quad 2 \quad 1]$ | $L_2(\bar{a}_2')$ | $\bar{a}_2' = [1 \quad 1 \quad 0.5]$ |
| $L_3(\bar{a}_3)$ | $\bar{a}_3 = [-1 \quad 1 \quad 3]$ | $L_3(\bar{a}_3')$ | $\bar{a}_3' = [-1.5 \quad 1.5 \quad 4.5]$ |

The input of the source data and the target data is shown in Fig. 3.




(a) (b)

Fig. 3. Input data. (a) Source data. (b) Target data

There are 1500 instances in the source domain and 15 instances in the target domain. Fig. 3(b) shows that there is good partition for the target data. To show the differences between the source data and the target data, the input data in both domains is displayed in Fig. 4, and the 3-dimension points in the form of input-output pairs are displayed in Fig. 5.

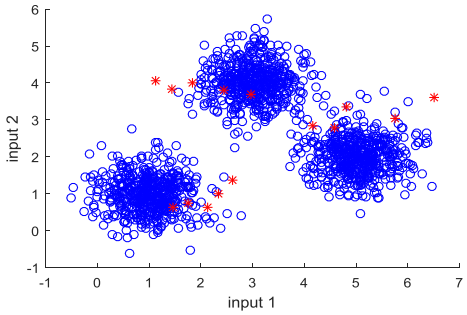



Fig. 4. Source (circle) and target (asterisk)

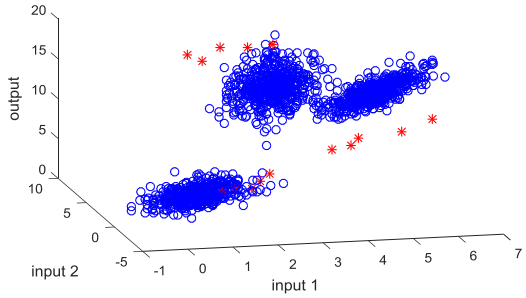



Fig. 5. Source (circle) and target (asterisk)(input-output)

From Fig. 4 and Fig. 5, we see that the input and output of the source data and the target data have different distributions.

In this experiment, PSO is applied to optimize the parameters and build the model $M'$ for the target domain. Parameters in PSO are as follows: the number of the initial population is 200, the maximum iteration is 200, the inertia weighting factor $w$ is 0.8, and two auxiliary parameters determining the dynamics of the population $c1$ and $c2$ are equal to 2. We apply a 5-fold cross validation procedure to construct the models. Table VII outlines the experimental results.

TABLE VII
THE RESULTS OF EXPERIMENT 1

| | Mean±Standard deviation |
|---|---|
| $Q$ | 0.00±0.00 |
| $Q_1$ | 32.30±0.06 |
| $Q_2$ | 0.00±0.00 |
| $Q_3$ | 4.00±3.40 |

From the results listed in Table VII, we can see that although few data are available in the target domain, they are still able to train a very good regression model $\bar{M}$, and the MSE of $\bar{M}$ on the testing set ($Q_2$) is 0.00. However, $Q_3$ is much larger than $Q_2$ (4.00>0.00), which indicates that knowledge from the source domain cannot improve the performance of the target domain, and is regarded as noise that harms the model's construction for the target domain.

Experiment 2

The purpose of this experiment is to apply and compare two optimization algorithms, namely PSO and DE, and to optimize the parameters of the mappings and build the new fuzzy regression transfer learning model $M'$ for the target domain. Further, the validation of the new method is confirmed.

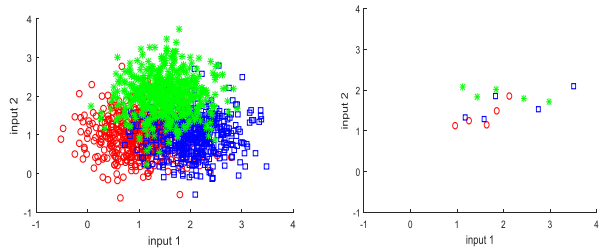
PSO and DE are computational methods that determine an optimal solution by iteratively navigating a population of solutions, which minimizes a certain predetermined objective function (performance index). There are three parameters in PSO that demonstrate significant impact on optimization performance: the inertia weight factor $w$, and two auxiliary parameters determining the dynamics of the population, $c1$ and $c2$. Typically, $w$, $c1$, and $c2$ assume value coming from several ranges, specifically $w \in [0.4,1.2]$, $c1 \in [1.4,2]$, $c2 \in [1.4,2]$), whereas $c1$ is equal to $c2$ [35]. In DE, optimization performance is largely dependent upon the values of the differential weight $F$ and the crossover probability $CR$. The value range of $F$ is [0,2], and the value range of $CR$ is [0,1] [36]. Based on the complexity of the problems, we apply the same initialization strategy in PSO and DE for all the experiments below; 200 candidate solutions are generated, and the maximum number of iterations is set to 200.

The distributions of the datasets applied in this experiment are shown in Table VIII.

TABLE VIII
DISTRIBUTIONS OF SOURCE DATA AND TARGET DATA

| Source data | | Target data | |
|---|---|---|---|
| Mean values | Covariance | Mean values | Covariance |
| $\boldsymbol{\mu}_1 = [1\ 1]$ | $\boldsymbol{\sigma}_1 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$ | $\boldsymbol{\mu}_1' = [1.5\ 1.5]$ | $\boldsymbol{\sigma}_1' = \begin{bmatrix} 0.4^2 & 0.1 \\ 0.1 & 0.4^2 \end{bmatrix}$ |
| $\boldsymbol{\mu}_2 = [2\ 1]$ | $\boldsymbol{\sigma}_2 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$ | $\boldsymbol{\mu}_2' = [2\ 1.5]$ | $\boldsymbol{\sigma}_2' = \begin{bmatrix} 0.5^2 & 0.1 \\ 0.1 & 0.5^2 \end{bmatrix}$ |
| $\boldsymbol{\mu}_3 = [1.5\ 2]$ | $\boldsymbol{\sigma}_3 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$ | $\boldsymbol{\mu}_3' = [2\ 2]$ | $\boldsymbol{\sigma}_3' = \begin{bmatrix} 0.5^2 & 0.1 \\ 0.1 & 0.5^2 \end{bmatrix}$ |

The linear functions in the source and target domains are the same as those in Experiment 1 in Table VI. The input of the source data and the target data is displayed in Fig. 6. As can be seen, there are some crossover regions between the source data and the target data. There are 1500 instances in the source domain, and 15 instances in the target domain.




(a) (b)

Fig. 6. Input data. (a) Source data. (b) Target data

To see the difference between the source data and target data more clearly, we combine them in one figure (Fig. 7). The 3-dimension points of the source data and the target data in the form of input-output pairs are drawn in Fig. 8.

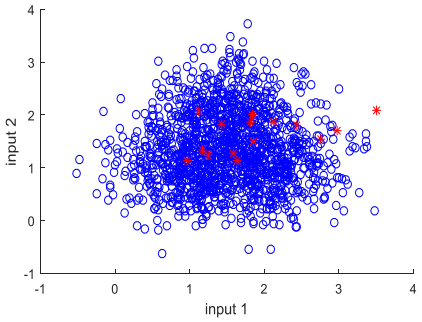



Fig. 7. Source (circle) and target (asterisk)

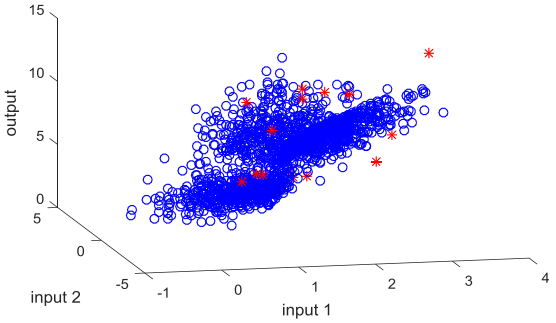



Fig. 8. Source (circle) and target (asterisk)(input-output)

We see from Figs. 6-8 that the distributions of both the input and output of the source data and the target data are different.

The results of $Q, Q_1$ and $Q_2$ are shown in Table IX.

TABLE IX
RESULTS OF $Q_1$, $Q_2$ AND $Q_3$

| | Mean ± Standard deviation |
|---|---|
| $Q$ | $0.01 \pm 0.00$ |
| $Q_1$ | $7.23 \pm 0.12$ |
| $Q_2$ | $183.80 \pm 405.14$ |

In the process of constructing a new fuzzy model $M'$ for the target domain to obtain $Q_3$, the experiment includes two parts to analyze the stability of the PSO and DE algorithms, and the generalization of the new fuzzy regression transfer learning model.

(a) The stability of the PSO and DE algorithms is analyzed.

The stability of PSO in different $w$, $c1$, and $c2$ is tested, and similarly, the stability of DE in different $F$ and $CR$ is tested. For comparison, the same source and target datasets in Table VIII are applied, and the experimental results of the new fuzzy regression transfer learning model on the target domain ($Q_3$) are displayed in Table X. Each experiment is run 10 times and we report the mean and standard deviation to quantify the stability of the solutions and the performance of the method.

TABLE X
THE STABILITY OF PSO AND DE

| PSO | | DE | |
|---|---|---|---|
| $w, c1 = c2$ | $Q_3$ | $F, CR$ | $Q_3$ |
| 0.6, 1.4 | $2.07 \pm 0.86$ | 0.5, 0.4 | $1.75 \pm 0.15$ |
| 0.6, 1.7 | $2.24 \pm 1.30$ | 0.5, 0.6 | $1.59 \pm 0.11$ |
| 0.6, 2 | $1.84 \pm 0.89$ | 0.5, 0.9 | $1.66 \pm 0.19$ |
| 0.9, 1.4 | $1.84 \pm 1.04$ | 1, 0.4 | $1.741 \pm 0.21$ |
| 0.9, 1.7 | $2.19 \pm 1.21$ | 1, 0.6 | $1.77 \pm 0.25$ |
| 0.9, 2 | $2.35 \pm 1.05$ | 1, 0.9 | $1.71 \pm 0.50$ |
| 1.2, 1.4 | $1.66 \pm 0.73$ | 1.5, 0.4 | $1.62 \pm 0.43$ |
| 1.2, 1.7 | $2.30 \pm 1.26$ | 1.5, 0.6 | $1.61 \pm 0.32$ |
| 1.2, 2 | $2.81 \pm 1.42$ | 1.5, 0.9 | $1.74 \pm 0.37$ |

We observe from Table X that the optimization performance of DE is better than that of PSO. Furthermore, the standard deviation in DE is smaller than the standard deviation in PSO, so the algorithmic stability of DE is superior to that of PSO.

(b) The generalization of the new fuzzy regression transfer learning model is studied.

Five-fold cross validation is applied in all the experiments. The dataset is split into five subsets, four of which are chosen as the training set, while the remaining subset forms the testing set. There are consequently five results for each model. The standard deviation of the five results indicates the generalization of the constructed model. The large standard deviation indicates either overfitting, or indicates that the data characteristics forming the training set do not coincide with the nature of the testing set. Conversely, the low value of the standard deviation shows that the model constructed on the basis of the training data has good generalization capability.

The generalization of the newly constructed model for the target domain is tested. PSO and DE with varying parameters are used to construct the model, and each experiment is run 10 times. The experimental results are shown in Table XI.

From the results, we observe that the values of the standard deviation, which reflects the generalization aspects of the constructed model, are not always very low. We claim that this situation is commonly encountered in transfer learning. Our assumption in the transfer learning problem is that the data in the target domain are limited in number and insufficient to

develop a good model. Therefore, there is only a small set of training data with which to construct a new fuzzy regression model for the target domain, and an even smaller set of testing data. The high values of the standard deviation in the 5-fold cross validation are anticipated, and clearly, these would decrease as the number of data in the target data increased. However if the size of the target data increases, the target data themselves could achieve the formation of a good model, with no need to transfer knowledge from another domain. The high values of the standard deviation in the 5-fold cross validation procedure are thus reasonable and to be expected.

TABLE XI
THE VALUES OF $Q_3$ UNDER DIFFERENT PARAMETERS

| PSO | | DE | |
|---|---|---|---|
| $w, c1 = c2$ | $Q_3$ | $F, CR$ | $Q_3$ |
| 0.6, 1.4 | 2.39±3.03 | 0.5, 0.4 | 2.32±1.51 |
| 0.6, 1.7 | 3.92±5.91 | 0.5, 0.6 | 2.14±1.44 |
| 0.6, 2 | 3.75±5.82 | 0.5, 0.9 | 2.16±1.51 |
| 0.9, 1.4 | 2.86±3.18 | 1, 0.4 | 2.46±1.83 |
| 0.9, 1.7 | 2.91±3.13 | 1, 0.6 | 2.36±1.81 |
| 0.9, 2 | 4.34±6.27 | 1, 0.9 | 2.40±1.77 |
| 1.2, 1.4 | 2.92±3.07 | 1.5, 0.4 | 2.31±1.81 |
| 1.2, 1.7 | 3.10±3.57 | 1.5, 0.6 | 2.42±1.82 |
| 1.2, 2 | 4.77±6.76 | 1.5, 0.9 | 2.27±1.73 |

In all the experiments in (a) and (b), the values of $Q_3$ are always smaller than the values of $Q_1$ and $Q_2$, and this demonstrates that the new fuzzy regression transfer leaning model $M'$is better than the existing model $M$ and the model $\bar{M}$ trained using few data in the target domain.

Experiment 3

This experiment investigates the impact of different input space reconstruction methods when there are interactions between the features. As there are two datasets in the source domain and target domain in transfer learning problems, we consider the following four cases in Table XII.

TABLE XII
FOUR CASES OF FEATURE INTERACTION

| | Source domain | Target domain |
|---|---|---|
| Interaction between features | N | N |
| | Y | Y |
| | Y | N |
| | N | Y |

Here, "N" indicates that there is no interaction between the features, and "Y" stresses that there is interaction between the features.

In the synthetic datasets, we generate data in the following way to highlight interaction scenarios between the features. The synthetic datasets are all two-dimensional. If the functions in the conclusion part of the fuzzy rules are in the form $L = a_0 + a_1x_1 + a_2x_2 + a_3x_1x_2$, we suppose that there is interaction between features $x_1$ and $x_2$. If we consider linear functions in the form $L = a_0 + a_1x_1 + a_2x_2$, we suppose that there is no interaction between the features.

We apply the two methods to construct the mappings for the input space with the structures shown in Fig. 9. In structure 1, the nonlinear mapping is constructed for each input variable. In structure 2, the nonlinear mapping is constructed for the entire input space (viz. all variables).

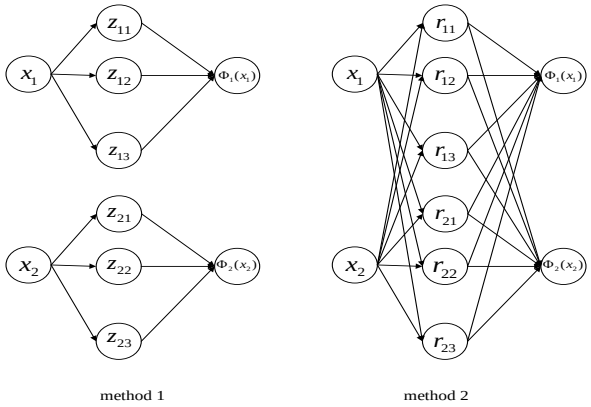



Fig. 9. Two methods of constructing the mappings for the input space

These two methods of constructing mappings for the input space are applied to the above four cases and the results are compared to determine whether there is an interaction between the features, and which method of transforming the input space is superior.

The same datasets as used in Experiment 2 are considered and every experiment is run 10 times. The experimental results of the proposed fuzzy regression transfer learning model in the target domain are reported in Table XIII.

TABLE XIII
SUMMARY OF THE EXPERIMENTAL RESULTS

| | $Q_3$(Mean ± Standard deviation) | |
|---|---|---|
| Cases | method 1 | method 2 |
| (N, N) | 0.51±0.08 | 2.41±1.78 |
| (Y, Y) | 0.35±0.13 | 1.77±3.55 |
| (Y, N) | 0.29±0.06 | 2.42±4.95 |
| (N, Y) | 0.48±0.04 | 0.98±0.54 |

From the results, we note that in all cases, whether or not there is an interaction between the features in the source domain and target domain, the performance of the first method is far better than the performance of the second method. Therefore, in a real-world problem where it is unknown whether there are interactions between features, it is advisable to use the first method to construct the transformation of the input space, i.e., construct the mapping for each input variable.

### *B. Real-world Datasets*

In this paper, we apply our fuzzy regression transfer learning method on two public datasets.

*1) Housing dataset*

The UCI Machine Learning Repository is a public dataset for regression problems. We use the "Housing Data Set" from this repository and revise it for the purpose of transfer learning. The dataset is split into two datasets using the attribute "TAX" to represent the full-value property-tax rate per $10,000. Instances

of "TAX" being smaller than 600 form the source dataset, and instances of "TAX" being larger than 600 constitute the target dataset. There are 360 instances in the source domain and 60 instances in the target domain. We select two attributes to construct the feature spaces (average number of rooms per dwelling, and weighted distance to five Boston employment centers), and use the attribute "MEDV", representing the median value of owner-occupied homes in $1000's, as the output value. The input data in the source and target domains are shown in Fig. 10, and the 3-dimensional input-output data points are displayed in Fig. 11.

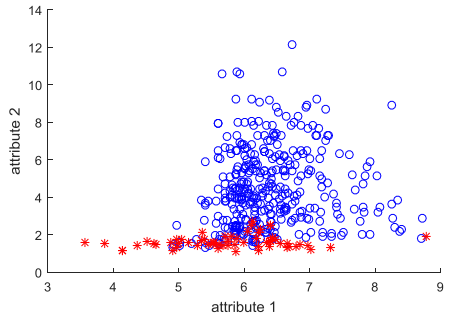



Fig. 10. Source (circle) and target (asterisk)

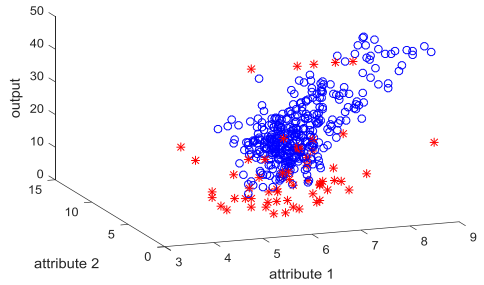



Fig. 11. Source (circle) and target (asterisk)(input-output)

As can be seen from Fig. 10 and Fig. 11, the distributions of both the input data and the output data in the two domains are different. The DE optimization algorithm is used to optimize the parameters, and $F = 0.5$, and $CR = 0.9$. This experiment also uses a 5-fold cross validation procedure, and for the purpose of analysis, the number of clusters is fixed as 6. The results are outlined in Table XIV, and the values of $Q$, $Q_1$, $Q_2$ and $Q_3$ are listed in Table XV.

As shown in Table XV, the values of $Q_1$ are all greater than 130, which indicates that the model for the source domain is unable to effectively solve regression tasks in the target domain. The results shown in the fourth column of the table indicate that the value of $Q_2$ is unstable. This is due to the small amount of available target data, so a model built from these training data cannot be generalized for the testing data.

The mean value of $Q_3$ shown in Table XIV is not small (171.70), but when analyzing the values of $Q_3$ in Table XV, we find that the other values of $Q_3$ are small and almost all are less than the values of $Q_1$ and $Q_2$. The results for the second experiment show an exception (572.06), but also lead to a large standard deviation. The reason for the large value of $Q_3$ is that the number of data in the target domain is small, so the training data does not reflect the characteristics of the testing data. Our new fuzzy regression transfer learning method therefore performs well on this real-world problem.

TABLE XIV
EXPERIMENTAL RESULTS OF THE REAL-WORLD DATASET "HOUSING"

| | Mean±Standard deviation |
|---|---|
| $Q$ | 16.77±9.03 |
| $Q_1$ | 144.53±13.35 |
| $Q_2$ | 26915.57±46894.50 |
| $Q_3$ | 171.70±224.09 |

TABLE XV
THE VALUES OF $Q_1$, $Q_2$ AND $Q_3$

| | $Q$ | $Q_1$ | $Q_2$ | $Q_3$ |
|---|---|---|---|---|
| 1 | 16.22 | 140.45 | 94.25 | 70.18 |
| 2 | 13.79 | 167.55 | 108388.78 | 572.06 |
| 3 | 30.68 | 133.81 | 170.59 | 75.25 |
| 4 | 17.44 | 143.39 | 31.22 | 86.26 |
| 5 | 5.69 | 137.48 | 25892.99 | 54.75 |

*2) Concrete compressive strength dataset*

In this series of experiments, we include another real-world dataset from the Machine Learning Repository, "Concrete Compressive Strength" data. The data has 8 attributes: "cement", "blast furnace slag", "fly ash", "water," "superplasticizer", "coarse aggregate", "fine aggregate", and "age", and the output feature is "concrete compressive strength". We revise the dataset in two aspects to make it more appropriate for transfer learning. First, the dataset is split into a source domain and a target domain according to the attribute "age": instances with "age" smaller than 100 fall into the source domain, and instances with "age" bigger than 100 fall into the target domain. To clearly differentiate between the source data and the target data, the attributes "blast furnace slag", "fly ash" and "superplasticizer" are perturbed by random numbers following different distributions in two domains. There are 900 instances in the source domain and 60 instances in the target domain.

For this dataset, DE is used to optimize the parameters, and $F = 0.5$, and $CR = 0.9$. In addition, we apply the method incorporating the automatic change of number of clusters described in Section III-A, and compare the results to choose the best solution. The 5-fold cross validation procedure is applied, and the results are reported in the form of *mean±standard deviation* in Table XVI.

TABLE XVI
EXPERIMENTAL RESULTS

| | 5 clusters | 6 clusters | 7 clusters | 8 clusters |
|---|---|---|---|---|
| Q | 0.02±0.01 | 0.02±0.00 | 0.02±0.00 | 0.02±0.00 |
| $Q_1$ | 1.97±1.22 | 2.65±1.27 | 2.20±0.94 | 2.13±1.00 |
| $Q_2$ | 116232.88±25981.71 | 161546.09±21531 6.00 | 1240.89±2648.12 | 249.13±409.39 |
| $Q_3$ | 0.01±0.00 | 0.01±0.00 | 0.01±0.00 | 0.01±0.00 |

High values of $Q_1$ indicate that the model in the source domain does not work well for the target data. High values of $Q_2$ indicate that the target data is insufficient for training a good model. From the results shown in the last row of Table XVIII, we see that the introduced fuzzy regression transfer learning method is effective in all cases, no matter how many clusters are used. When the number of clusters is set to 5, the mean value of $Q_3$ in 5-fold cross validation is the smallest, as is the standard deviation. As a result, using 5 clusters for this problem is the best option.

## V. Conclusion and Future Work

In this study, we proposed a fuzzy regression transfer learning method that modifies the input space of data through mappings to make the fuzzy rules of the existing model more compatible for solving tasks in the target domain. This method effectively solves regression problems in the target domain when only a small amount of data is available. Experimental results show that our method greatly improves the performance of the existing model in estimating the values of the target domain.

We have only considered situations in which the input spaces of the source domain and the target domain respectively have the same dimensionality. In further studies, it will be beneficial to focus on the more challenging problem in which the two domains have different feature spaces.

## Acknowledgment

The work presented in this paper was supported by the Australian Research Council (ARC) under Discovery Grant DP140101366.

## References

[1] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowledge-Based Systems*, vol. 80, pp. 14-23, 2015.

[2] Q. Xu and Q. Yang, "A survey of transfer and multitask learning in bioinformatics," *Journal of Computing Science and Engineering*, vol. 5, pp. 257-268, 2011.

[3] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345-1359, 2010.

[4] S. J. Pan, V. W. Zheng, Q. Yang, and D. H. Hu, "Transfer learning for wifi-based indoor localization," in *Proceedings of the Workshop on Transfer Learning for Complex Tasks of the 23rd AAAI Conference on Artificial Intelligence*, 2008.

[5] Z. Sun, Y. Chen, J. Qi, and J. Liu, "Adaptive localization through transfer learning in indoor wi-fi environment," in *Proceedings of the 7th International Conference on Machine Learning and Application*, pp. 331-336, 2008.

[6] Q. Liu, X. Liao, H. Li, J. R. Stack, and L. Carin, "Semisupervised multitask learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 1074-1086, 2009.

[7] H. Zuo, G. Zhang, V. Behbood, J. Lu, and X. Meng, "Transfer Learning in Hierarchical Feature Spaces," *10th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, pp. 183-188, Nov. 2015.

[8] J. Yang, R. Yan, and A. G. Hauptmann, "Cross-domain video concept detection using adaptive SVMS," in *Proceedings of the 15th International Conference on Multimedia*, pp. 188-197, 2007.

[9] Q. Yang, Y. Chen, G.-R. Xue, W. Dai, and Y. Yu, "Heterogeneous transfer learning for image clustering via the social web," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, vol. 1, pp. 1-9, 2009.

[10] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proceedings of the 24th International Conference on Machine Learning*, pp. 193-200, 2007.

[11] A. Storkey, "When training and test sets are different: characterizing learning transfer," *Dataset Shift in Machine Learning*, pp. 3-28, 2009.

[12] H. Zuo, G. Zhang, V. Behbood and J. Lu, "Feature Spaces-based Transfer Learning", IFSA World Congress and EUSFLAT Annual Meeting (IFSA-EUSFLAT), 2015.

[13] L. Duan, I. W. Tsang, and D. Xu, "Domain transfer multiple kernel learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 465-479, 2012.

[14] T. G. Dietterich, P. Domingos, L. Getoor, S. Muggleton, and P. Tadepalli, "Structured machine learning: The next ten years," *Machine Learning*, vol. 73, pp. 3-23, 2008.

[15] S. Bickel, M. Brückner, and T. Scheffer, "Discriminative learning for differing training and test distributions," in *Proceedings of the 24th International Conference on Machine* Learning, pp. 81-88, 2007.

[16] J. Huang, A. Gretton, K. M. Borgwardt, B. Schölkopf, and A. J. Smola, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems*, pp. 601-608, 2006.

[17] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Self-taught clustering," in *Proceedings of the 25th International Conference on Machine Learning*, pp. 200-207, 2008.

[18] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pp. 677-682, 2008.

[19] Z. Wang, Y. Song, and C. Zhang, "Transferred dimensionality reduction," in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases Part II*, pp. 550-565, 2008.

[20] L. Borzemski and G. Starczewski, "Application of transfer regression to TCP throughput prediction," in *Proceedings of the First Asian Conference on Intelligent Information and Database Systems*, pp. 28-33, 2009.

[21] P. Yang, Q. Tan, and Y. Ding, "Bayesian task-level transfer learning for non-linear regression," in *Proceedings of the International Conference on Computer Science and Software* Engineering, pp. 62-65, 2008.

[22] Z. Chen, S. Aghakhani, J. Man, and S. Dick, "ANCFIS: A neurofuzzy architecture employing complex fuzzy sets," *IEEE Transactions on Fuzzy Systems*, vol. 19, pp. 305-322, 2011.

[23] A. Lemos, W. Caminhas, and F. Gomide, "Multivariable Gaussian evolving fuzzy modeling system," *IEEE Transactions on Fuzzy Systems*, vol. 19, pp. 91-104, 2011.

[24] X. Yang, G. Zhang, J. Lu, and J. Ma, "A kernel fuzzy c-means clustering based fuzzy support vector machine algorithm for classification problems with outliers or noises," *IEEE Transactions on Fuzzy Systems*, vol. 19, pp. 105-115, 2011.

[25] V. Behbood, J. Lu, and G. Zhang, "Fuzzy bridged refinement domain adaptation: Long-term bank failure prediction," *International Journal of Computational Intelligence and Applications*, vol. 12, p. 1350003, 2013.

[26] V. Behbood, J. Lu, and G. Zhang, "Fuzzy refinement domain adaptation for long term prediction in banking ecosystem," *IEEE Transactions on Industrial Informatics*, vol. 10, pp. 1637-1646, 2014.

[27] V. Behbood, J. Lu, G. Zhang, and W. Pedrycz, "Multi- step fuzzy bridged refinement domain adaptation algorithm and its application to bank failure prediction," *IEEE Transactions on Fuzzy Systems*, vol. 23, pp. 1917-1935, 2015.

[28] Z. Deng, Y. Jiang, K.-S. Choi, F.-L. Chung, and S. Wang, "Knowledge-leverage-based TSK fuzzy system modeling," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, pp. 1200-1212, 2013.

[29] Z. Deng, Y. Jiang, F.-L. Chung, H. Ishibuchi, and S. Wang, "Knowledge-leverage-based fuzzy system and its modeling," *IEEE Transactions on Fuzzy Systems*, vol. 21, pp. 597-609, 2013.

[30] H. Zuo, G. Zhang, V. Behbood, J. Lu, W. Pedrycz, and T. Zhang, "Fuzzy Transfer Learning in Data-Shortage and Rapidly Changing Environments," in *Proceedings of the 12th International FLINS Conference (FLINS 2016)*, pp. 175-180, 2016.

[31] J. Shell and S. Coupland, "Fuzzy transfer learning: Methodology and application," *Information Sciences*, vol. 293, pp. 59-79, 2015.

[32] W. Pedrycz, *Granular Computing: Analysis and Design of Intelligent Systems*, CRC Press, 2013.

[33] W. Pedrycz, B. Russo, and G. Succi, "Knowledge transfer in system modeling and its realization through an optimal allocation of information granularity," *Applied Soft Computing*, vol. 12, pp. 1985-1995, 2012.
[34] S. Das, A. Abraham, and A. Konar, "Particle swarm optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives," in *Advances of Computational Intelligence in Industrial Systems*, Springer, pp. 1-38, 2008.
[35] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *Evolutionary Programming VII*, 1998, pp. 591-600.
[36] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer Science & Business Media, 2006.

**Hua Zuo** is working toward the Ph.D. degree with the Faculty of Engineering and Information Technology, University of Technology Sydney, Australia.

Her research interests include transfer learning and fuzzy systems.

She is a Member of the Decision Systems and e-Service Intelligence (DeSI) Research Laboratory, Center for Quantum Computation and Intelligence Systems, University of Technology Sydney.

**Guangquan Zhang** is an Associate Professor and Co-Director of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. He received the Ph.D. degree in applied mathematics from Curtin University of Technology, Perth, Australia, in 2001.

His research interests include fuzzy sets and systems, fuzzy optimization, fuzzy transfer learning, and fuzzy modelling in machine learning and data analytics. He has authored four monographs, five textbooks, and 300 papers including 154 refereed international journal papers.

Dr. Zhang has won six Australian Research Council (ARC) Discovery Projects grants and many other research grants. He was awarded ARC QEII fellowship in 2005. He has served as a member of the editorial boards of several international journals, as a guest editor of eight special issues for IEEE transactions and other international journals, and co-chaired several international conferences and workshops in the area of fuzzy decision-making and knowledge engineering.

**Witold Pedrycz** (F'98) is Professor and Canada Research Chair (CRC) in Computational Intelligence in the Department of Electrical and Computer Engineering, University of Alberta, Canada. He received the PhD and DSci from the Silesian University of Technology, Poland. He is a foreign member of the Polish Academy of Sciences, a Fellow of the Royal Society of Canada. He received a prestigious Norbert Wiener award from the IEEE Systems, Man, and Cybernetics Society, the IEEE Canada Computer Engineering Medal, a Cajastur Prize for Soft Computing from the European Centre for Soft Computing, a Killam Prize, and a Fuzzy Pioneer Award from the IEEE Computational Intelligence Society.

His main research directions involve computational intelligence, fuzzy modeling and granular computing, and data mining. He is an author of 15 research monographs and numerous papers in international journals and conferences.

He is the Editor-in-Chief of ***Information Sciences (Elsevier)***, ***WIREs Data Mining and Knowledge Discovery*** (Wiley), and ***International Journal of Granular Computing*** (Springer); and an Associate Editor of ***IEEE Transactions on Fuzzy Systems***.

**Vahid Behbood** received the Ph.D. degree in software engineering from the University of Technology Sydney, Australia, in 2013.

He is currently working as a Lecture with the Faculty of Engineering and Information Technology, University of Technology Sydney. His research interests include machine learning, fuzzy sets and systems, data analytic and warning systems. He has published more than 20 papers in international journals and conferences.

**Jie Lu** (SM'13) is a Professor and the Associate Dean Research with the Faculty of Engineering and Information Technology at the University of Technology Sydney, Australia. She received the Ph.D. degree in information systems from Curtin University of Technology, Australia, in 2000.

Her main research expertise is in decision support systems, recommender systems, fuzzy transfer learning, concept drift and their applications in e-business.

She has published 10 research books and over 400 papers in refereed journals and conference proceedings with over 150 papers in IEEE Transactions and other international journals; awarded 7 Australian Research Council (ARC) Discovery Project grants and many other research grants. She is a member of the ARC College.

She serves as Editor-In-Chief for Knowledge-Based Systems (Elsevier), Editor-In-Chief for International Journal on Computational Intelligence Systems (Atlantis), Associate Editor for IEEE Trans on Fuzzy Systems, Editor for book series on Intelligent Information Systems (World Scientific), and has served as a guest editor of 12 special issues for IEEE transactions and other international journals, general/PC/organization chairs for ten international conferences as well as having delivered 14 keynote/ plenary speeches at IEEE and other international conferences.