Fuzzy Rule-based Domain Adaptation in Homogeneous and Heterogeneous Spaces

Hua Zuo, Jie Lu, *Fellow, IEEE*, Guangquan Zhang, and Witold Pedrycz, *Fellow, IEEE*

***Abstract*— Domain adaptation aims to leverage knowledge acquired from a related domain (called a source domain) to improve the efficiency of completing a prediction task (classification or regression) in the current domain (called the target domain), which has a different probability distribution from the source domain. Although domain adaptation has been widely studied, most existing research has focused on homogeneous domain adaptation, where both domains have identical feature spaces. A new challenge, proposed in recent years in this area, is heterogeneous domain adaptation, in which both the probability distributions and the feature spaces are different. Moreover, in both homogeneous and heterogeneous domain adaptation, the greatest efforts and the major achievements have been made with classification tasks, while successful solutions for tackling regression problems are limited. This paper presents two innovative fuzzy rule-based methods to deal with regression problems. The first method, called FHoDA, handles homogeneous spaces, and the second method, called FHeDA, handles heterogeneous spaces. Fuzzy rules are first generated from the source domain through a learning process; these rules, also known as knowledge, are then transferred to the target domain by establishing a latent feature space to minimize the gap between the feature spaces of the two domains. Through experiments on synthetic datasets, we demonstrate the effectiveness of both methods and discuss the impact of some of the significant parameters that affect performance. Experiments on real-world datasets also show that the proposed methods improve the performance of the target model over an existing source model or a model built using a small amount of target data.**

***Index Terms*—Domain adaptation, fuzzy rules, machine learning, transfer learning, regression**

I. Introduction

HIGH volume, high velocity, and high variety (the three Vs) are the key features of big data. The rapid evolution of data has left traditional machine learning methods far behind. Rapidly changing data can lead to the failure of a learning model if the distribution of the current data differs too greatly from the training data. However, simply discarding the existing model is wasteful and training a new model from scratch is both time-consuming and labor-intensive. Moreover, learning a new model requires a massive amount of data with responses, which may not always be available, especially if the subject area is new.

As a possible solution to a lack of enough data with responses, transfer learning [1] aims to construct new predictive models much more quickly and effectively by exploiting the knowledge accumulated in an auxiliary domain. Some examples include: using already-categorized French documents to help classify English documents [2]; building recognition models capable of identifying novel visual categories [3]; and detecting a user’s current location given previously collected WiFi data [4].

Transfer learning sits within the machine learning research area; hence, its methods use many notable machine learning techniques as basic training models, such as SVM [5, 6], neural networks [7, 8], naïve Bayes [9, 10], and case-based models [11, 12]. For more information, we refer the reader to several survey papers that provide reviews and summaries of the various transfer learning methods and categories [13-15].

The methods proposed in this paper aim to solve two types of transfer learning problems that fall within the category of homogeneous and heterogeneous domain adaptation. Within the homogeneous domain adaptation methods, the representative methods include transfer component analysis (TCA) [4], the geodesic flow kernel (GFK) approach [16], information-theoretical learning (ITL) [17], sampling geodesic flow (SGF) [18], transfer deep network (TDN) [8], feature-level domain adaptation (FLDA) [19], and scatter component analysis (SCA) [20]. Within the heterogeneous domain adaptation methods, the typical methods are heterogeneous spectral mapping (HeMap) [21], alignment-based models (MA) [22], semi-supervised kernel matching for domain adaptation (SSKMDA) [2], the DASH-N model [23], and kernel canonical correlation analysis (KCCA) [24].

These methods have had some success in handling domain adaptation issues but ignore the inherent phenomenon of uncertainty – a crucial factor during the knowledge transfer process. There is a clear co-dependency between the level of certainty in learning a task and the amount of information that is available; problems with too little information have a high degree of uncertainty. If there are too few data with responses in the target domain, only a finite amount of information can be extracted, and this leads to a high degree of uncertainty.

Manuscript received September 18, 2017; revised April 30, 2018; accepted July 3, 2018. This work was supported by the Australian Research Council under DP 140101366.

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

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

However, the introduction of fuzzy systems has shown promising results in overcoming this problem.

Behbood et al. [25, 26] proposed a fuzzy-based transductive transfer learning approach to long-term bank failure prediction models with differing data distributions in the source and target domains. They first applied a fuzzy neural network to predict the initial labels for data in the target domain, then used fuzzy similarity measures to refine the labels. To improve performance, they simultaneously took similarity and dissimilarity into account during the refinement process. Using fuzzy techniques to measure the similarity, the authors revealed the advantage of fuzzy logic in knowledge transfer when the target domain lacks critical information, is vague, and involves uncertainty. Deng et al. [27-30] proposed a series of transfer learning methods based on a Mamdani-Larsen-type fuzzy system and a Takagi-Sugeno-Kang TSK fuzzy model to deal with the insufficient data scenarios by integrating with the corresponding knowledge-leverage mechanism. Further, their methods were applied to recognize the electroencephalogram signals in a data shortage environment.

Most existing transfer learning methods are intended for classification tasks, yet only a few concentrate on regression problems. Some of our previous work looked at solving regression tasks in homogeneous domain adaptation problems based on fuzzy rule models. In [31, 32], we proposed a granular fuzzy regression domain adaptation method for transfer learning in regression tasks. It contains three algorithms, each designed to solve a different domain adaptation case: 1) where the fuzzy rule conditions differ; 2) where the fuzzy rule conclusions differ; and 3) where both differ. In case 1, Algorithm 1 changes the input space by constructing a mapping for each input variable using a network with sigmoid functions. In case 2, Algorithm 2 modifies the output space with the mappings built in the same way as the input space. In case 3, Algorithm 3 changes both the input and the output spaces. All the work in these papers have two limitations: 1) they have an assumption that both domains had the same number of fuzzy rules. 2) they only deal with homogeneous domain adaptation.

Obviously, the assumption of an identical number of fuzzy rules in two domains is too rigid in practice, and existing methodologies for homogeneous settings cannot be directly applied to heterogeneous settings.

To solve these two limitations, we first develop a method to deal with the homogeneous domain adaptation where the number of fuzzy rules in the source domain and target domain do not need to be the same. The fuzzy rules of the source domain are used and modified to fit the target data, and target data without responses are applied to improve the performance of the target model. Further, we consider a more challenging problem – heterogeneous domain adaptation. The fuzzy rules acquired from the source domain are transferred to the target domain by extracting a latent feature space to minimize the gap between the feature spaces of the two domains.

The main contributions of this paper are twofold. First, we propose a fuzzy homogeneous domain adaptation (FHoDA) method that performs transfer learning in homogeneous spaces where, although the feature space is the same in both domains, the probability distributions of the input variables are different. Second, we present a fuzzy heterogeneous domain adaptation (FHeDA) method that handles knowledge transfer in heterogeneous spaces, where both the feature space and the probability distributions are different in the source and target domains. To the best of our knowledge, this is the first paper to deal with heterogeneous domain adaptation problems in a fuzzy rule-based system.

The remainder of this paper is structured as follows. Section II provides the preliminaries used in this paper, including some important definitions in transfer learning and the basic fuzzy rule-based model applied in our method – the Takagi-Sugeno fuzzy model. Section III illustrates the domain adaptation problems we aim to solve in this work from a fuzzy rule-based model perspective. Section IV details the domain adaptation methods and related algorithms for homogeneous spaces, followed by the heterogeneous spaces. In Sections V and VI, synthetic and real-world datasets are used to validate the proposed methods. The final section concludes the paper and outlines future work.

II. Preliminaries

The definitions of transfer learning, homogeneous domain adaptation, and heterogeneous domain adaptation, are given in this section, followed by a description of the Takagi-Sugeno fuzzy model.

A. *Definitions*

Definition 1 (Domain) [1]: A domain is denoted by $D=\{F,P(X)\}$, where F is a feature space, and $P(X)$, $X=\{x_1,\cdots,x_n\}$, are the probability distributions of the instances.

Definition 2 (Task) [1]: A task is denoted by $T=\{Y,f(\cdot)\}$, where $Y\in R$ is the response, and $f(\cdot)$ is an objective predictive function.

Definition 3 (Transfer Learning) [1]: Given a source domain D_s, a learning task T_s, a target domain D_t, and a learning task T_t, transfer learning aims to improve learning of the target predictive function $f_t(\cdot)$ in D_t using the knowledge in D_s and T_s where $D_s\neq D_t$ or $T_s\neq T_t$.

Transfer learning addresses the problem of how to leverage previously acquired knowledge (a source domain) to improve the efficiency of learning in a new domain (the target domain).

Definition 4 (Homogeneous Domain Adaptation) [33]: Homogeneous domain adaptation is a category of transfer learning in which the feature space is the same $F_t=F_s$, but the corresponding probability distributions are different $P_t(X)\neq P_s(X)$.

Definition 5 (Heterogeneous Domain Adaptation) [33]: Heterogeneous domain adaptation is a category of transfer learning in which the feature spaces are different $F_t\neq F_s$, and the corresponding probability distributions are different $P_t(X)\neq P_s(X)$.

B. *Takagi-Sugeno Fuzzy Model*

The basic model applied in this paper is the Takagi-Sugeno fuzzy model [34, 35]. It is composed of c fuzzy rules with the following representation:

If $\boldsymbol{x}$ is $A_i(\boldsymbol{x}, \boldsymbol{v}_i)$, then y is $L_i(\boldsymbol{x}, \boldsymbol{a}_i)$ $\quad i = 1, \dots, c$ (1)

$\boldsymbol{v}_i$ are the porotypes, and $\boldsymbol{a}_i$ are the coefficients of the linear functions.

The output of the Takagi-Sugeno fuzzy model is calculated by

$$y = \sum_{i=1}^{c} \frac{A_i(x, v_i)}{\sum_{j=1}^{c} A_j(x, v_j)} \cdot L_i(\boldsymbol{x}, \boldsymbol{a}_i) \tag{2}$$

This fuzzy rule-based model is built using a set of instances $\{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$ using a sequence of two procedures [36]:

Step 1: Build the condition parts of the rules using a fuzzy clustering algorithm.

The conditions of the fuzzy rules $A_1, \dots, A_c$ are constructed using the fuzzy C-means (FCM) algorithm [37]. This clustering algorithm is applied to divide the input data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$ into c clusters, and obtain the centers of the clusters $\boldsymbol{v}_1, \dots, \boldsymbol{v}_c$. Each cluster defines a fuzzy set. The corresponding membership functions of the fuzzy sets are

$$A_i(\boldsymbol{x}, \boldsymbol{v}_i) = 1 / \sum_{j=1}^{c} \left(\frac{\|x - v_i\|}{\|x - v_j\|} \right)^{\frac{2}{m-1}} \qquad i = 1, \dots, c \tag{3}$$

where $m\ (m > 1)$ is a fuzzification coefficient that affects both the shape and overlap of the resulting membership functions. (3) defines the membership degree of instance $\boldsymbol{x}$ belonging to the i-th fuzzy set (cluster).

Step 2: Construct the linear functions in the conclusions of the rules.

The vector of parameters $\boldsymbol{a}$ of the linear functions can be derived and optimized using a set of data with responses. For a given input, the Takagi-Sugeno fuzzy model's output is parameter $\boldsymbol{a}$'s linear function, and the optimal $\boldsymbol{a}$ can be calculated analytically [31]:

$$\boldsymbol{a} = (\boldsymbol{F}^T \boldsymbol{F})^{-1} \boldsymbol{F}^T \boldsymbol{y} \tag{4}$$

where $\boldsymbol{a} = [\boldsymbol{a}_1 \cdots \boldsymbol{a}_c]^{\boldsymbol{T}}$, $\boldsymbol{F} = [\boldsymbol{f}(\boldsymbol{x}_1)^T \cdots \boldsymbol{f}(\boldsymbol{x}_N)^T]^{\boldsymbol{T}}$, $\boldsymbol{f}(\boldsymbol{x}_k)^T = \begin{bmatrix} A_1(\boldsymbol{x}_k, \boldsymbol{v}_1) & \dots & A_c(\boldsymbol{x}_k, \boldsymbol{v}_c) \\ A_1(\boldsymbol{x}_k, \boldsymbol{v}_1)\boldsymbol{x}_k & \dots & A_c(\boldsymbol{x}_k, \boldsymbol{v}_c)\boldsymbol{x}_k \end{bmatrix}$, $k = 1, \cdots, N$, and $\boldsymbol{y} = [y_1 \cdots y_N]^{\boldsymbol{T}}$.

Based on steps 1 and 2, the conditions and conclusions of the fuzzy rules are formed, and a Takagi-Sugeno fuzzy model is built.

III. Problem Statement

The dataset in the source domain is denoted by $\boldsymbol{D} = \{(\boldsymbol{x}_1^s, y_1^s), \cdots, (\boldsymbol{x}_{N_s}^s, y_{N_s}^s)\}$, where $\boldsymbol{x}_k^s \in R^{n_s}$, $k = 1, \cdots, N_s$ is the n_s-dimensional input variable, the response $y_k^s \in R$ is the continuous output variable, and N_s indicates the number of data. Since the amount of source data with responses is massive, a well-performing regression model for the source domain – the Takagi-Sugeno fuzzy model M_s – can be learned.

model M_s

if $\boldsymbol{x}^s$ is $A_i(\boldsymbol{x}^s, \boldsymbol{v}_i^s)$, then y^s is $L_i(\boldsymbol{x}^s, \boldsymbol{a}_i^s)$ $\quad i = 1, \cdots, c_s$ (5)

The dataset in the target domain contains two subsets: one with responses and one without responses $\boldsymbol{H} = \{\boldsymbol{H}_L, \boldsymbol{H}_U\} = \{\{(\boldsymbol{x}_1^t, y_1^t), \cdots, (\boldsymbol{x}_{N_{t1}}^t, y_{N_{t1}}^t)\}, \{\boldsymbol{x}_{N_{t1+1}}^t, \cdots, \boldsymbol{x}_{N_t}^t\}\}$, where $\boldsymbol{x}_k^t \in R^{n_t}$, $k = 1, \cdots, N_t$ is the n_t-dimensional input variable, $y_k^t \in R$, $k = 1, \cdots, N_{t1}$ is the continuous output variable. $\boldsymbol{H}_L$ includes the instances with responses, and $\boldsymbol{H}_U$ contains the data without responses. The number of data in $\boldsymbol{H}_L$ and $\boldsymbol{H}_U$ are N_{t1} and $N_t - N_{t1}$ respectively, and satisfy $N_{t1} \ll N_t$, $N_{t1} \ll N_s$.

Suppose the ideal model for the target domain is M_t.

model M_t

if $\boldsymbol{x}^t$ is $A_i(\boldsymbol{x}^t, \boldsymbol{v}_i^t)$, then y^t is $L_i(\boldsymbol{x}^t, \boldsymbol{a}_i^t)$ $\quad i = 1, \cdots, c_t$ (6)

Building a well-performing Takagi-Sugeno fuzzy model needs a large amount of data with responses, and inadequate data in $\boldsymbol{H}_L$ cannot guarantee the constructed model in the target domain will perform well. Furthermore, discrepancies between the source and target data mean that using the source model to solve target tasks is impossible.

Clarifying the divergence between the source and target data plays a crucial role in conducting knowledge transfers from the source domain to the target domain. Since the proposed methods aim to adapt the domains using fuzzy rule-based models, we differentiate the source domain and target domain using the feature space and fuzzy rules. In general, the difference between the source and target domains is summarized by the four cases shown in Fig. 1.

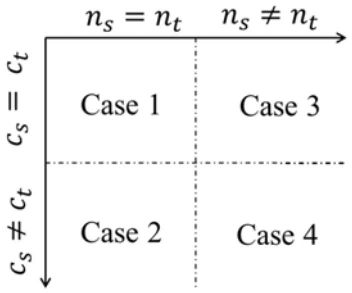


Fig. 1. Four cases distinguishing the source and target domains

(1) Case 1: $n_s = n_t$, and $c_s = c_t$. The input spaces (feature spaces) in the source and target domains have the same dimensionality with different distributions, and the number of constructed fuzzy rules is also equal in both domains.

(2) Case 2: $n_s = n_t$, and $c_s \neq c_t$. The input spaces in both domains have the same dimension with different distributions, but the number of fuzzy rules is different.

(3) Case 3: $n_s \neq n_t$, and $c_s = c_t$. Discrepancies in the input data in both domains occur in dimensionality and in distribution. However, there are an equal number of fuzzy rules in both domains.

(4) Case 4: $n_s \neq n_t$ and $c_s \neq c_t$. This is the most complicated case. The input space, in dimensionality and in distribution, and the number of fuzzy rules are both different in the source and target domains.

Based on the definition of domain adaptation, Case 1 and Case 2 belong to homogeneous domain adaptation, and Case 3 and Case 4 fall into the scope of heterogeneous domain adaptation.

IV. Fuzzy Domain Adaptation in Homogeneous and Heterogeneous Spaces

This section presents the fuzzy rule-based methods and corresponding algorithms for domain adaptation completed in homogeneous and heterogeneous spaces.

A. *Homogeneous Domain Adaptation*

Case 1, typical homogeneous domain adaptation problems were addressed in our previous paper [32]. We considered three different situations and proposed corresponding algorithms to deal with each of them.

This section concentrates on Case 2, homogeneous domain adaptation problems where the two domains have a different number of fuzzy rules. Both situations are addressed: where the number of fuzzy rules in the source domain is greater than in the target domain, and vice versa.

When the source domain has a greater number of fuzzy rules, the source model can still be used and modified to fit the target data because Takagi-Sugeno fuzzy models weight some linear functions in a nonlinear way when fitting to a curve. Each linear function represents a fuzzy rule, and all the fuzzy rules form a fuzzy partition of the output space. The greater the number of fuzzy rules, the more precise the partition of the space is. It is, therefore, reasonable to consider that a Takagi-Sugeno fuzzy model can approximate any curve as long as there are an adequate number of fuzzy rules. And, if there are a greater number of fuzzy rules in the source domain than in the target domain, then it is reasonable to revise the fuzzy rules in the source domain to fit the target data.

When the target domain has a greater number of fuzzy rules, the fuzzy rules in the source domain must be reconstructed to a number that is no less than the target domain. Then, the rebuilt fuzzy rules are modified and applied to fit the target data.

Further, in addition to target data with responses, target data without responses are also used to improve the performance of the target model. We assume that instances that are close to each other in the input space will have similar responses. The closer the instances, the more similar their responses.

Our fuzzy homogeneous domain adaptation (FHoDA) method to deal with knowledge transfer with fuzzy rule-based models follows. The FHoDA method contains two main steps.
Step 1: Train the source model.

A source model M_s is built based on the source dataset $\boldsymbol{D}$.
model M_s

$$\text{if } \boldsymbol{x}^s \text{ is } A_i(\boldsymbol{x}^s, \boldsymbol{v}_i^s)\text{, then } y^s \text{ is } L_i(\boldsymbol{x}^s, \boldsymbol{a}_i^s) \qquad i = 1, \cdots, c \tag{7}$$

where $c = \max(c_s,\ c_t)$.

$c = \max(c_s,\ c_t)$ ensures that the number of trained fuzzy rules are sufficient to fit the target data later. So, in the first situation, c is simply equal to c_s. But in the second situation, c is equal to c_t, which means that the source model is reconstructed with the same number of fuzzy rules as the target domain. This can facilitate modification and transfer of the fuzzy rules between domains.
Step 2: Modify the existing fuzzy rules to construct the target model.

There can be two possible differences between the fuzzy rules in the source and target domains: their conditions and/or their conclusions. However, because the number of fuzzy rules is not equal, it is hard to tell exactly where the differences are – the conditions or the conclusions. To guarantee the best results, we modify the existing model using the three algorithms proposed in our previous paper [32] and select the one with the best performance as the target model, i.e., we change the input space, change the output space, and change both spaces.

The corresponding target models are:
model M_{t1} (changes of input space)

$$\text{if } \boldsymbol{x}^t \text{ is } A_i(\Phi(\boldsymbol{x}^t), \Phi(\boldsymbol{v}_i^s))\text{, then } y^t \text{ is } L_i(\Phi(\boldsymbol{x}^t), \boldsymbol{a}_i^s)$$
$$i = 1, \cdots, c \tag{8}$$

model M_{t2} (changes of output space)

$$\text{if } \boldsymbol{x}^t \text{ is } A_i(\boldsymbol{x}^t, \boldsymbol{v}_i^s)\text{, then } y^t \text{ is } \Psi_i(L_i(\boldsymbol{x}^t, \boldsymbol{a}_i^s))$$
$$i = 1, \cdots, c \tag{9}$$

model M_{t3} (changes of input and output space)

$$\text{if } \boldsymbol{x}^t \text{ is } A_i(\Phi(\boldsymbol{x}^t), \Phi(\boldsymbol{v}_i^s))\text{, then } y^t \text{ is } \Psi_i(L_i(\Phi(\boldsymbol{x}^t), \boldsymbol{a}_i^s))$$
$$i = 1, \cdots, c \tag{10}$$

where $\Phi = [\Phi_1 \cdots \Phi_n]$, $n = n_s = n_t$, and $\Psi = [\Psi_1 \cdots \Psi_c]$ are the transformation mappings for the input space and output space.

The final target model M_t is chosen from the best among models M_{t1}, M_{t2} and M_{t3}, i.e.,

$$M_t = M_{ti}\text{, if } M_{ti} \geq M_{tj}\ , i,j = 1,2,3 \tag{11}$$

where $M_{ti} \geq M_{tj}$ means the performance of M_{ti} on the target data $\boldsymbol{H}_U$ is no worse than that of M_{tj}.

The construction of mappings Φ and Ψ is the key element in the FHoDA method. The nonlinear functions are used to build the mappings. The nonlinear function is constructed through a network that is composed of P nodes in the hidden layer and a single node in the output layer. The transformation of the jth input variable of data $\boldsymbol{x}_k^t$ is shown in Fig. 2 as an example of the nonlinear mapping for each input variable.

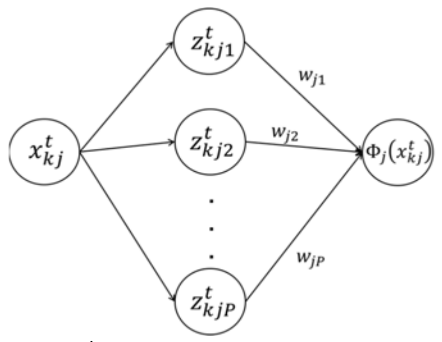


Fig. 2. Nonlinear mapping structure

The active functions of the nodes in the hidden layer are sigmoid functions, which are dominated by two parameters. Therefore, as shown in Fig. 2, the graphical representation of the transformed jth input variable of data $\boldsymbol{x}_k^t$ is:

$$\Phi_j\left(\boldsymbol{x}_{kj}^t\right) = \sum_{p=1}^{P} w_{jp} * \boldsymbol{z}_{kj}^t \tag{12}$$

where w_{jp} indicates the weights of the pth node's contribution to the output, and $\boldsymbol{z}_{kj}^t = \frac{1}{1+e^{-\alpha_{jp}(x_{kj}^t-\beta_{jp})}}$, $j = 1, \ldots, n$, $p = 1, \ldots, P$, $\alpha_{jp} > 0$.

The mappings for the output space are constructed in the same way. The output of each rule is modified by a nonlinear function with the structure shown in Fig. 2, and finally the entire output space is changed.

The parameters of the mappings Φ and Ψ are obtained through an optimization procedure, but the cost functions are different when optimizing Φ and Ψ to get models M_{t1}, M_{t2} and M_{t3}.

When training model M_{t1}, i.e., applying the algorithm that changes the input space, the cost function is

$$\begin{aligned} S1 &= \sqrt{\frac{1}{N_{t1}} \sum_{k=1}^{N_{t1}} \left(\sum_{i=1}^{c} \frac{A_i(\Phi(\boldsymbol{x}_k^t), \Phi(\boldsymbol{v}_i^s))}{\sum_{j=1}^{c} A_j(\Phi(\boldsymbol{x}_k^t), \Phi(\boldsymbol{v}_j^s))} L_i(\Phi(\boldsymbol{x}_k^t), \boldsymbol{a}_i^s) - y_k^t \right)^2} \\ &+ \lambda_1 \sqrt{\frac{1}{N_{t1} * h} \sum_{k=1}^{N_{t1}} \sum_{l=1}^{h} (y_k^t - y_k^t(l))^2 * \exp(-\|x_k^t - x_k^t(l)\|)} + \frac{\lambda_2}{2} w^T w \end{aligned} \tag{13}$$

where $y_k^t(l) = \sum_{i=1}^{c} \frac{A_i(\Phi(x_k^t(l)), \Phi(\boldsymbol{v}_i^s))}{\sum_{j=1}^{c} A_j(\Phi(x_k^t(l)), \Phi(\boldsymbol{v}_j^s))} L_i(\Phi(x_k^t(l)), \boldsymbol{a}_i^s)$.

The first term in the cost function (13) trains the model based on the target data with responses, which aims to minimize the gap between the output of the constructed model and the target data's real response. The second term operates on the assumption that data with less distance in the input space will have a similar response. Therefore, for each target data $\boldsymbol{x}_k^t$ in $\boldsymbol{H}_L$, the h-nearest data $\{\boldsymbol{x}_k^t(1) \cdots \boldsymbol{x}_k^t(h)\}$ in $\boldsymbol{H}_U$ are found, and the outputs of $\{\boldsymbol{x}_k^t(1) \cdots \boldsymbol{x}_k^t(h)\}$ are expected to be close to the response of $\boldsymbol{x}_k^t$. $\exp(-\|\boldsymbol{x}_k^t - \boldsymbol{x}_k^t(l)\|$ determines that the data that are closer to the center $\boldsymbol{x}_k^t$, will have an output more approximate to the response of $\boldsymbol{x}_k^t$. The third term is a structural risk of the cost function, and parameter λ_2 indicates a trade-off between the quality of an approximation and the complexity of the approximation function. w is the vector of all the optimized parameters.

When training model M_{t2}, i.e., applying the algorithm that changes the output space, the cost function is

$$S2 = \sqrt{\frac{1}{N_{t1}} \sum_{k=1}^{N_{t1}} \left(\sum_{i=1}^{c} \frac{A_i(\boldsymbol{x}_k^t, \boldsymbol{v}_i^s)}{\sum_{j=1}^{c} A_j(\boldsymbol{x}_k^t, \boldsymbol{v}_j^s)} \Psi_i(L_i(\boldsymbol{x}_k^t, \boldsymbol{a}_i^s)) - y_k^t \right)^2} + \frac{\lambda_2}{2} w^T w \tag{14}$$

The cost function for training the mappings for the output space contains two terms. Both are the same as the first and third term in $S1$ – one trains the model with the target data with responses, the other restrains the complexity of the approximation function. But here, we do not use target data without responses to train the target model. This is because $\boldsymbol{x}_k^t$ needs to find the h -nearest data based on distance in the input space to best use data without responses in the target domain. However, the algorithm that changes the output space focuses on modifying the output variables; therefore, using the data without responses here may have a negative impact on the model's construction.

When training model M_{t3}, i.e., applying the algorithm that changes the input and output spaces, the cost function is

$$S3 = \sqrt{\frac{1}{N_{t1}} \sum_{k=1}^{N_{t1}} \left(\sum_{i=1}^{c} \frac{A_i(\Phi(\boldsymbol{x}_k^t), \Phi(\boldsymbol{v}_i^s))}{\sum_{j=1}^{c} A_j(\Phi(\boldsymbol{x}_k^t), \Phi(\boldsymbol{v}_j^s))} \Psi_i(L_i(\Phi(\boldsymbol{x}_k^t), \boldsymbol{a}_i^s)) - y_k^t \right)^2} + \frac{\lambda_2}{2} w^T w \tag{15}$$

Similarly, only the target data with responses are applied to train the mappings for the input and output spaces simultaneously. And a structural risk is added to the cost function to control the complexity of the approximation function.

The overall algorithm for the FHoDA method described above is provided in Algorithm 1.

Algorithm 1. Homogeneous Domain Adaptation Procedure

```
Input: $\boldsymbol{D}$, $\boldsymbol{H}$,
Output: $\boldsymbol{Y}_U$ for $\boldsymbol{H}_U$

1. Train source model $M_s$ based on $\boldsymbol{D}$
2. Modify the fuzzy rules in $M_s$
    2.1 Change input space, and get model $M_{t1}$
    2.2 Change output space, and get model $M_{t2}$
    2.3 Change both input and output spaces, and get model $M_{t3}$
3. Compare $M_{t1}$, $M_{t2}$, and $M_{t3}$, and choose the best one as $M_t$
4. Use $M_t$ to predict the response $\boldsymbol{Y}_U$ for $\boldsymbol{H}_U$
```

B. Heterogeneous Domain Adaptation

The greatest challenge in heterogeneous domain adaptation is the different dimensions of the input spaces in both domains. This means that the distributions of the input variables are not only different but also the number of the input variables. To eliminate the gap caused by the mismatch of the feature spaces, many studies in heterogeneous domain adaptation employ a method that extracts a latent feature space that is shared between both domains. This space can then be used to facilitate knowledge exploration and transfer between the domains. After projecting the input data of the two domains into the latent feature space, the heterogeneous domain adaptation problem is converted to a homogeneous domain adaptation problem. In our method, we also use an extraction approach.

As discussed in Section III, Cases 3 and 4 both belong to the category of heterogeneous domain adaptation. The distinction between them is the number of the fuzzy rules in the two domains. In Case 3, the number of fuzzy rules is equal; in Case 4, they are not. After transforming the data from the original feature space to the latent feature space, some information will be lost, and we cannot guarantee the number of fuzzy rules will remain unchanged in the latent feature space. However, the relation is not limited by the number of fuzzy rules in the domains. As long as there are a sufficient number of constructed fuzzy rules in the latent feature space of the source domain, they can be modified and used for tasks in the target domain.

Therefore, the proposed fuzzy heterogeneous domain adaptation (FHeDA) method for solving Cases 3 and 4 follows with no discrimination. The FHeDA method contains three steps for transferring knowledge from the source domain to the target domain.

Step 1: Extract the latent feature space and map all the input data to it.

Since the primary factor impeding knowledge transfer across the domains is a mismatched feature space, the first step is to map the source and target data into a uniform feature space, where common features can benefit from the discovery and transfer of the knowledge. This minimizes the gap between the distributions of the input variables for both domains. Approximating the input data distributions across both domains has two benefits:

a. The conditions of the fuzzy rules are dominated by the center of the clusters, which are derived from the input data using a clustering algorithm. In turn, the relative location of the center of the clusters greatly influences the construction of the fuzzy rules. Converting the input data into a common latent feature space forces the data distribution in each domain to approximate the other, which reduces the difference between the relative location of the center of the clusters in both domains.

b. Projecting to the latent feature space also facilitates knowledge transfer of the fuzzy rule conclusions, which are represented as the linear functions of input variables. Similar distributions restrict the input variables in approximate ranges and reduce disagreement in the input data in both domains.

In this paper, the canonical correlation analysis (CCA) algorithm [36] has been used to derive the latent feature space. The latent feature space is extracted by learning a mapping between the original feature space and the latent feature space. Based on the input data $\{\boldsymbol{x}_1^s, \cdots, \boldsymbol{x}_{N_s}^s\}$ and $\{\boldsymbol{x}_1^t, \cdots, \boldsymbol{x}_{N_t}^t\}$, two mappings U_s and U_t are learned simultaneously to convert the input data from the original feature spaces of two domains to the latent feature space.

Under the mappings U_s and U_t, the input data in two domains will have a new representation as follows:

$$U_s(\boldsymbol{x}_k^s) = \overline{\boldsymbol{x}}_k^s, k = 1, \cdots, N_s \quad (16)$$
$$U_t(\boldsymbol{x}_k^t) = \overline{\boldsymbol{x}}_k^t, k = 1, \cdots, N_t \quad (17)$$

Therefore, the input data $\{\boldsymbol{x}_1^s, \cdots, \boldsymbol{x}_{N_s}^s\}$ and $\{\boldsymbol{x}_1^t, \cdots, \boldsymbol{x}_{N_t}^t\}$ in two domains becomes $\{\overline{\boldsymbol{x}}_1^s, \cdots, \overline{\boldsymbol{x}}_{N_s}^s\}$ and $\{\overline{\boldsymbol{x}}_1^t, \cdots, \overline{\boldsymbol{x}}_{N_t}^t\}$ in the latent feature space, and the dimension of the latent feature space is $\bar{n} = \min(n_s, n_t)$.

Step 2: Build a Takagi-Sugeno fuzzy model for the source domain in the latent feature space.

The dataset $\boldsymbol{D}$ in the source domain has become $\overline{\boldsymbol{D}} = \{(\overline{\boldsymbol{x}}_1^s,\ y_1^s), \cdots, (\overline{\boldsymbol{x}}_{N_s}^s,\ y_{N_s}^s)\}$, and a Takagi-Sugeno fuzzy model $\overline{M}_s$ is built for the source domain in the latent feature space.

model $\overline{M}_s$

$$\text{if } \overline{\boldsymbol{x}}^s \text{ is } A_i(\overline{\boldsymbol{x}}^s, \overline{\boldsymbol{v}}_i^s), \text{ then } y^s \text{ is } L_i(\overline{\boldsymbol{x}}^s, \overline{\boldsymbol{a}}_i^s) \qquad i = 1, \cdots, \bar{c} \quad (18)$$

where $\bar{c}$ is the critical parameter that determines the amount of fuzzy rules in the source domain and in the target domain. This parameter is so pivotal, Section V presents a set of experiments designed to explore the impact of $\bar{c}$ on the model's construction.

Although the discrepancy between the input data's distributions in the two domains has been reduced in the latent feature space, a gap still exists and cannot be completely eliminated. Moreover, different linear functions (the conclusions of the fuzzy rules) are another factor that distinguish the source and target domains, so model $\overline{M}_s$ in the source domain cannot be directly used to solve regression tasks in the target domain.

Step 3: Modify the existing fuzzy rules in model $\overline{M}_s$ to make them suitable for the target data.

In the latent feature space, it is difficult to detect which parts of the two domains' fuzzy rules are different. Using the same strategy as in homogeneous domain adaptation, we modify the source model in three different ways and choose the one with the best performance on the target data.

model $\overline{M}_{t1}$(changes of the input space)

$$\text{if } \overline{\boldsymbol{x}}^t \text{ is } A_i(\Phi(\overline{\boldsymbol{x}}^t), \Phi(\overline{\boldsymbol{v}}_i^s)), \text{ then } y^t \text{ is } L_i(\Phi(\overline{\boldsymbol{x}}^t), \overline{\boldsymbol{a}}_i^s)$$
$$i = 1, \cdots, \bar{c} \quad (19)$$

model $\overline{M}_{t2}$ (changes of the output space)

$$\text{if } \overline{\boldsymbol{x}}^t \text{ is } A_i(\overline{\boldsymbol{x}}^t, \overline{\boldsymbol{v}}_i^s), \text{ then } y^t \text{ is } \Psi_i(L_i(\overline{\boldsymbol{x}}^t, \overline{\boldsymbol{a}}_i^s))$$
$$i = 1, \cdots, \bar{c} \quad (20)$$

model $\overline{M}_{t3}$(changes of input and the output space)

$$\text{if } \overline{\boldsymbol{x}}^t \text{ is } A_i(\Phi(\overline{\boldsymbol{x}}^t), \Phi(\overline{\boldsymbol{v}}_i^s)), \text{ then } y^t \text{ is } \Psi_i(L_i(\Phi(\overline{\boldsymbol{x}}^t), \overline{\boldsymbol{a}}_i^s))$$
$$i = 1, \cdots, \bar{c} \quad (21)$$

where $\Phi = [\Phi_1 \cdots \Phi_{\bar{n}}]$, and $\Psi = [\Psi_1 \cdots \Psi_{\bar{c}}]$ are the transformation mappings for the input space and output space.

The final target model $\bar{M}_t$ is chosen from the best among the models $\bar{M}_{t1}$, $\bar{M}_{t2}$ and $\bar{M}_{t3}$, i.e.,

$$\bar{M}_t = \bar{M}_{ti}, \text{ if } \bar{M}_{ti} \geq \bar{M}_{tj}\ ,\ i,j = 1,2,3 \tag{22}$$

where $\bar{M}_{ti} \geq \bar{M}_{tj}$ means the performance of $\bar{M}_{ti}$ on the target data $\boldsymbol{H}_U$ is no worse than $\bar{M}_{tj}$.

The construction of the mappings for the input and output spaces is the same as for homogeneous domain adaptation, i.e., using a network to modify each input or output variable. The parameters of the mappings are derived by minimizing the following cost functions.

When changing the input space to get model $\bar{M}_{t1}$, the cost function below is minimized

$$W1 = \sqrt{\frac{1}{N_{t1}} \sum_{k=1}^{N_{t1}} \left(\sum_{i=1}^{\bar{c}} \frac{A_i(\Phi(\bar{\boldsymbol{x}}_k^t),\Phi(\bar{\boldsymbol{v}}_i^s))}{\sum_{j=1}^{\bar{c}} A_j(\Phi(\bar{\boldsymbol{x}}_k^t),\Phi(\bar{\boldsymbol{v}}_j^s))} L_i(\Phi(\bar{\boldsymbol{x}}_k^t), \bar{\boldsymbol{a}}_i^s) - y_k^t\right)^2} + \frac{\lambda_2}{2} w^T w \tag{23}$$

The cost function includes two terms: one aims to decrease the gap between the output of the constructed target model and the data's real responses, and the other is a structural risk term to control the complexity of the built model. Here, only the target data with responses are applied to modify the existing model. The reason target data without responses is not used is that the transformation from the original feature space to the latent feature space may change the manifold of the input space. The data that is close in distance in the latent feature space may be far from each other in the original feature space. So using neighboring target data without responses in the latent feature space to improve the result is risky, and negatively impact the performance of the constructed model. This is checked experimentally in Section VI, A.

When changing the output space, the cost function $W2$ is minimized

$$W2 = \sqrt{\frac{1}{N_{t1}} \sum_{k=1}^{N_{t1}} \left(\sum_{i=1}^{\bar{c}} \frac{A_i(\bar{\boldsymbol{x}}_k^t,\bar{\boldsymbol{v}}_i^s)}{\sum_{j=1}^{\bar{c}} A_j(\bar{\boldsymbol{x}}_k^t,\bar{\boldsymbol{v}}_j^s)} \Psi_i(L_i(\bar{\boldsymbol{x}}_k^t, \bar{\boldsymbol{a}}_i^s)) - y_k^t\right)^2} + \frac{\lambda_2}{2} w^T w \tag{24}$$

Similarly, when changing the input and output space simultaneously, the cost function $W3$ is minimized

$$W3 = \sqrt{\frac{1}{N_{t1}} \sum_{k=1}^{N_{t1}} \left(\sum_{i=1}^{\bar{c}} \frac{A_i(\Phi(\bar{\boldsymbol{x}}_k^t),\Phi(\bar{\boldsymbol{v}}_i^s))}{\sum_{j=1}^{\bar{c}} A_j(\Phi(\bar{\boldsymbol{x}}_k^t),\Phi(\bar{\boldsymbol{v}}_j^s))} \Psi_i(L_i(\Phi(\bar{\boldsymbol{x}}_k^t), \bar{\boldsymbol{a}}_i^s)) - y_k^t\right)^2} + \frac{\lambda_2}{2} w^T w \tag{25}$$

The overall algorithm corresponding to the FHeDA method is provided in Algorithm 2.

Algorithm 2. Heterogeneous Domain Adaptation Procedure

```
Input: $\boldsymbol{D}$, $\boldsymbol{H}$,
Output: $\boldsymbol{Y}_U$ for $\boldsymbol{H}_U$

1. Use CCA to learn $U_s$ and $U_t$
2. Map $\boldsymbol{x}_k^s$ to $\bar{\boldsymbol{x}}_k^s$, and map $\boldsymbol{x}_k^t$ to $\bar{\boldsymbol{x}}_k^t$
3. Train model $\bar{M}_s$ using $\bar{\boldsymbol{D}}$
4. Modify the fuzzy rules in $\bar{M}_s$
    4.1 Change input space to get $\bar{M}_{t1}$
    4.2 Change output space to get $\bar{M}_{t2}$
    4.3 Change both input and output spaces to get $\bar{M}_{t3}$
5. Compare $\bar{M}_{t1}$, $\bar{M}_{t2}$, and $\bar{M}_{t3}$, and choose the best one as $\bar{M}_t$
6. Use $\bar{M}_t$ to predict the output $\boldsymbol{Y}_U$ for $\boldsymbol{H}_U$
```

V. Experiments in Homogeneous Domain Adaptation

The experiments reported in this section focus on homogeneous domain adaptation. Both synthetic and real-world datasets were used to validate the proposed method and explore the properties related to the method's performance.

A. Synthetic Datasets

The experiments on the synthetic datasets comprise three sections. The first section introduces the synthetic datasets and the experimental settings. The second section validates the effectiveness of the proposed FHoDA method. The final section analyzes the sensitivity of some critical parameters.

1) Datasets and Experimental Settings

Four synthetic datasets were generated with different numbers of fuzzy rules to simulate various cases of homogeneous domain adaptation. As shown in Table I, 500 instances were generated for each rule, so dataset 2r contained 1000 instances, dataset 3r contained 1500, and so on.

TABLE I
Four datasets with a different number of fuzzy rules

	number of fuzzy rules	number of instances
dataset 2r	2	1000
dataset 3r	3	1500
dataset 4r	4	2000
dataset 5r	5	2500

In each experiment, two of the four datasets were chosen as the source and the target domain respectively. All the data in the source domain had responses, but only 5% of the data in the target domain had responses. The remaining responses were only available during the testing procedure. In total, 12 experiments simulating homogeneous domain adaptation with the fuzzy rule-based models were executed.

The FHoDA method generates the target model through an optimization process. In this work, a differential evolution (DE) optimization algorithm was used to optimize the parameters of the constructed mappings. DE is a computational method that determines an optimal solution by iteratively navigating a population of solutions to minimize a certain predetermined

TABLE II
COMPARISON RESULTS OF DIFFERENT TRANSFER LEARNING METHODS

Source to target		RMSE of models						
	Baseline 1	Baseline 2	Baseline 3	TCA	SA	GFK	TSK-method	FHoDA
5r to 4r	5.23±0.00	5.07±0.00	101.85±21599.44	7.88±0.00	7.58±0.00	7.60±0.00	0.91±0.01	0.68±0.01
5r to 3r	3.67±0.00	3.62±0.00	14.85±900.34	4.66±0.00	4.65±0.00	5.16±0.00	1.18±0.07	1.14±0.05
5r to 2r	6.81±0.00	6.74±0.00	3.45±54.25	4.82±0.00	4.80±0.00	5.15±0.00	1.13±0.03	0.62±0.01
4r to 3r	0.97±0.00	0.96±0.00	17.90±828.50	2.19±0.00	2.37±0.00	2.61±0.00	0.70±0.00	0.65±0.00
4r to 2r	3.11±0.00	3.06±0.00	3.45±54.25	2.11±0.00	2.22±0.00	2.47±0.00	1.03±0.01	0.85±0.01
3r to 2r	3.81±0.00	3.74±0.00	3.45±54.25	3.28±0.00	3.23±0.00	3.19±0.00	0.98±0.05	0.50±0.04
2r to 3r	4.07±0.00	3.96±0.00	4.82±30.58	1.17±0.00	1.14±0.00	1.21±0.00	1.24±0.02	1.03±0.00
2r to 4r	3.45±0.00	3.26±0.00	43.29±6502.94	4.45±0.00	4.23±0.00	4.72±0.00	1.08±0.04	0.73±0.00
2r to 5r	7.65±0.00	6.50±0.03	47.83±10791.78	3.61±0.00	3.90±0.01	3.87±0.03	2.44±0.26	2.04±0.00
3r to 4r	0.94±0.00	0.94±0.00	30.12±2643.94	6.10±0.01	6.36±0.00	6.09±0.00	0.79±0.01	0.72±0.01
3r to 5r	4.16±0.00	3.87±0.00	500.06±979947.30	3.25±0.00	3.21±0.00	3.05±0.00	1.91±0.04	1.57±0.00
4r to 5r	4.67±0.00	4.43±0.02	47.73±10746.32	5.91±0.02	5.45±0.01	5.86±0.01	1.55±0.00	1.39±0.01

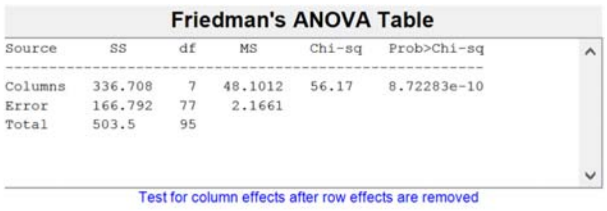
objective function. Such algorithms are commonly known as metaheuristics, as they make few, or no, assumptions about the problem being optimized and are able to search very large populations of candidate solutions. Beyond DE algorithms, particle swarm optimization (PSO) algorithms were also frequently used. However, based on the experimental results from our previous studies [31], their performance and stability on this class of problems is inferior, so we chose a DE algorithm to optimize the parameters of the transformation mappings and construct the target model. Five-fold cross validation was used for the construction, so all results are shown in the form "mean±variance".

2) *Regression Results*

As described above, 12 experiments simulating homogeneous domain adaptation with fuzzy rule-base models were conducted. The results are shown in Table II.

The left column in Table II indicates the source and target domains. For example, '5r to 4r' indicates that the source domain is 'dataset 5r', and the target domain is 'dataset 4r'. The second to the fourth columns are the three baselines of the transfer learning problem: 1) the root mean square error (RMSE) of the source model on the target data without responses $\boldsymbol{H}_U$; 2) the RMSE of the model trained using source and target data on $\boldsymbol{H}_U$; 3) the RMSE of the model trained using only target data on $\boldsymbol{H}_U$. The fifth to the seventh columns show the RMSE's of three famous transfer learning approaches (TCA, SA, and GFK) on $\boldsymbol{H}_U$, respectively. The results in the eighth column show the RMSE of a TSK-based fuzzy method on $\boldsymbol{H}_U$. And the final column shows the RMSE of our proposed method on $\boldsymbol{H}_U$.

First, a Friedman test was conducted on the RMSE of all the methods shown in Table II. The results shown in Fig.3 indicate that the performance of different transfer learning methods was statistically significant. Further, a pairwise comparison using a multiple comparison test was performed, and the results are shown in Table III.



```
Friedman's ANOVA Table
Source      SS       df     MS       Chi-sq    Prob>Chi-sq
------------------------------------------------------------
Columns   336.708     7   48.1012    56.17     8.72283e-10
Error     166.792    77    2.1661
Total     503.5      95
Test for column effects after row effects are removed
```

Fig. 3. Results of Friedman test

TABLE III
PAIRWISE COMPARISON RESULTS

		p-values
Baseline 1		0.0000
Baseline 2		0.0000
Baseline 3		0.0000
TCA	FHoDA	0.0000
SA		0.0000
GFK		0.0000
TSK-method		0.0110

In Table III, the first two columns indicate the compared two methods, and the last column shows the p-value for a hypothesis test where the corresponding mean difference is equal to zero. From the statistical analysis results, there is a significant difference between our method, the three baselines, the three state-of-the-art transfer learning methods, and the TSK-method.

Therefore, we can conclude that the performance of our method exceeds the three baselines, and is superior to the existing state-of-the-art non-fuzzy transfer learning approaches.

Moreover, since target data without responses $\boldsymbol{H}_U$ are used to improve model M_{t1}'s performance, Table IV compares the RMSE of model M_{t1} when built with and without target data that had no responses. Lower values are shown in bold.

TABLE IV
M_{t1} BUILT USING/NOT USING $\boldsymbol{H}_U$ – MODEL COMPARISON

Source to target datasets	RMSE of the models	
	model M_{t1} (not using $\boldsymbol{H}_U$)	model M_{t1}(using $\boldsymbol{H}_U$)
5r to 4r	1.0781± 0.0004	**1.0756± 0.0004**
5r to 3r	0.9352± 0.0057	**0.8962± 0.0083**
5r to 2r	0.5602± 0.0016	**0.5573± 0.0005**
4r to 3r	2.1269± 0.2059	**2.0996± 0.1718**
4r to 2r	1.5981± 0.0310	**1.5711± 0.0108**
3r to 2r	0.5882± 0.0016	**0.5772± 0.0005**
2r to 3r	0.8080± 0.0093	**0.8074± 0.0044**
2r to 4r	1.2522± 0.0009	1.2947± 0.0039
2r to 5r	3.6904± 0.0012	3.7228± 0.0002
3r to 4r	0.8876± 0.0009	**0.8457± 0.0005**
3r to 5r	2.5273± 0.0007	**2.5397± 0.0014**
4r to 5r	3.0755± 0.0110	**3.0614± 0.0037**

The results show that using target data $\boldsymbol{H}_U$ for training was better in ten of the twelve experiments – a clear performance improvement for M_{t1}. However, in two experiments, using target data without responses had a negative impact. The cause may lie in the model's construction. If some of the target data with responses were located at the junction of two clusters (fuzzy rules), the utilization of $\boldsymbol{H}_U$, finding the h-nearest data without responses for each target data with response, will result in an inappropriately constructed model.

3) *Parameter Sensitivity Analysis*

Within FHoDA's optimization procedure, some parameters play an important role in model construction in the target domain. The three groups of experiments, shown in Figs. 4-6, were designed to explore the impact of these parameters. Fig. 4 depicts the effect of parameter p (the number of nodes used to construct mappings for the input space) on model M_{t1}'s performance. Fig. 5 shows model M_{t2}'s performance with a varying q (the number of nodes used when building the mappings for the output space). Fig. 6 charts model M_{t1}'s performance with different values for h (the number of target data without responses selected for each target data with response). Only the results for the experiments '5r to 4r', '5r to 3r' and '5r to 2r' have been included in the figures to illustrate the impact of the parameters on the performance of the model.

Observing the results shown in Fig. 4, the parameter p had a slight impact on model construction in experiment '5r to 2r'. The variations in the model's performance as p varied were almost the same in experiments '5r to 4r' and '5r to 3r'. When p was changed from 2 to 4, the RMSE increased, then decreased with a p of 5, and peaked at a p of 6.

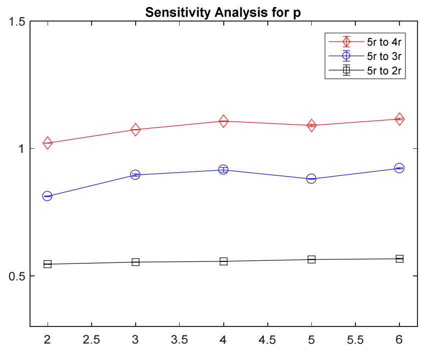


Fig. 4. Sensitivity analysis of parameter p in three experiments

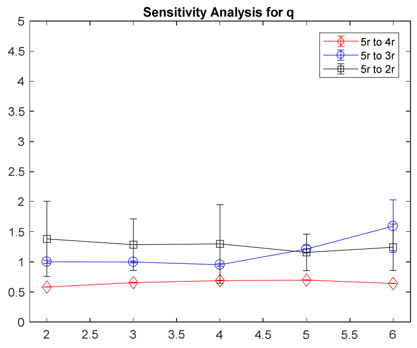


Fig. 5. Sensitivity analysis of parameter q in three experiments

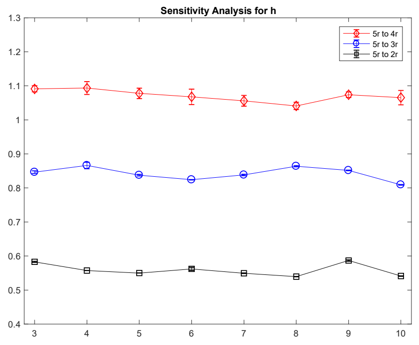


Fig. 6. Sensitivity analysis of parameter h in three experiments

The results in Fig. 5 show no obvious changes to model construction with different values of q in the '5r to 4r' experiment. In '5r to 3r', the RMSE shows a rising trend at q values greater than 4, and, in '5r to 2r', the RMSE fluctuates as the q values change.

Fig. 6 shows fluctuations in all three experiments. The model's best performance appears when h=8 in '5r to 4r', h =10 in '5r to 3r', and h =8 in '5r to 2r'. Notice that the performance of the model does not always develop an increasing trend. When more data without responses around the target data with responses are selected to improve model construction, data at greater distances are found. It is unreasonable to suppose that data at large distances will have similar responses, so a growing number of selected target data without responses will eventually lead to a negative impact on the model's optimization.

B. Real-world Datasets

Since the studies on regression problems of adapting a domain are scarce, there are no public datasets in these scenarios. In this work, therefore, the real-world datasets from UCI Machine Learning Repository are used and modified to simulate the regression domain adaptation problems and verify the proposed FHoDA method. The way of modifying the datasets is crucial, so a detailed description to these datasets is provided.

The "Physicochemical Properties of Protein Tertiary Structure" dataset contains nine features to predict the RMSD-size of the residue. The first 30,000 instances were chosen as the source domain, and the last 10,000 instances were used to form the target domain. To increase the level of discrepancy between the source and target data, features in the source data, "non-polar exposed area", "fractional area of exposed non-polar residue", and "molecular mass-weighted exposed area" were perturbed with random numbers using the normal distribution $\mathrm{N}(0.1,0.1)$. These three features in the target data, were perturbed with random numbers using the normal distributions $\mathrm{N}(7,1)$, $\mathrm{N}(5,1)$, and $\mathrm{N}(8,1)$, respectively. All the data in the source domain had responses, but only 100 data in the target domain had responses.

The "Housing" dataset aims to predict the "MEDV" (Median value of owner-occupied homes in \$1000's) using six input attributes. The data was normalized and split into two datasets using the attribute "TAX", which represents the full-value property-tax rate per \$10,000. Instances of "TAX" smaller than 0.5 were used to form the source dataset, and instances of "TAX" larger than 0.5 were used as the target dataset. The attributes "RM", "AGE", and "B" in the source data were perturbed by random numbers from $\mathrm{N}(0.1, 0.1)$, while those attributes in the target data were perturbed by normal random numbers using the distributions $\mathrm{N}(7,1)$, $\mathrm{N}(5,1)$, and $\mathrm{N}(8,1)$, respectively. There were 360 instances with responses in the source domain and 130 instances in the target domain; 15 had responses.

Although a target domain may only contain a small amount of data with responses, it can still be used to train a model. However, we assert that a model trained solely on a small amount of data with responses will not perform well. And, to support this assertion, we trained a target model $\widetilde{M}_t$ using target data with responses and tested its performance.

Since it is hard to get the information needed to indicate the number of fuzzy rules in real-world datasets, we used a brute-force approach of trying every number in a given range and selecting the one with the best performance. The RMSE of models M_s, $\widetilde{M}_t$, and M_t for the target data $\boldsymbol{H}_U$ are shown in Table V.

From the results in Table V, we can conclude that the source model does not fit the target data. The high mean value and RMSE variance of model $\widetilde{M}_t$ verifies our assumption that a model trained with little data shows poor performance. Model M_t performed much better than the source model M_s and $\widetilde{M}_t$, which was trained with less data.

TABLE V
EXPERIMENTAL RESULTS FOR FHoDA ON REAL-WORLD DATA

Protein tertiary structure				Housing			
c	M_s	$\widetilde{M}_t$	M_t	c	M_s	$\widetilde{M}_t$	M_t
8	50.88± 27.15	18.36± 21.07	6.00± 0.01	5	1.40± 0.71	0.31± 0.03	0.19± 0.00
9	48.90± 37.85	14.92± 3.34	5.93± 0.01	6	3.11± 0.41	0.65± 0.18	0.22± 0.01
10	43.32± 87.07	17.86± 5.32	6.10± 0.01	7	2.41± 0.21	0.34± 0.05	0.15± 0.00
11	36.84± 23.49	26.96± 37.27	5.90± 0.01	8	2.51± 0.25	0.20± 0.00	0.15± 0.01
12	54.41± 15.00	16.50± 20.65	5.98± 0.00	9	1.60± 1.14	0.21± 0.01	0.15± 0.00

VI. EXPERIMENTS IN HETEROGENEOUS DOMAIN ADAPTATION

This section describes the experiments in heterogeneous domain adaptation. The synthetic datasets were designed to analyze the proposed FHeDA method in a controlled environment. Real-world datasets were then used to validate the practicability of the proposed method.

A. Experiments on Synthetic Datasets

As with homogeneous domain adaptation, this section comprises three subsections. Information about the synthetic datasets and the experimental settings is provided, followed by the experimental results. The second subsection validates the effectiveness of the FHeDA method, and the third subsection discusses the impact of a critical parameter on the model's construction.

1) Datasets and Experimental Settings

Four datasets of different dimensions were generated according to different numbers of fuzzy rules to simulate various source and target domains in heterogeneous spaces. Information about the generated datasets is provided in Table VI. For example, 'dataset 2' contains 3-dimensional data, which was generated according to 4 fuzzy rules.

TABLE VI
HETEROGENEOUS DOMAIN ADAPTATION DATASETS

	dimension	number of fuzzy rules
dataset 1	3	3
dataset 2	3	4
dataset 3	4	3
dataset 4	4	4

TABLE VII
SETTINGS FOR THE EXPERIMENTS IN HETEROGENEOUS DOMAIN ADAPTATION

	Source domain	Target domain	dimension(S) vs dimension(T)	rules(S) vs rules (T)
Exp 1	dataset 1	dataset 3	3 vs 4	3 vs 3
Exp 2	dataset 1	dataset 4		3 vs 4
Exp 3	dataset 2	dataset 3		4 vs 3
Exp 4	dataset 3	dataset 1	4 vs 3	3 vs 3
Exp 5	dataset 3	dataset 2		3 vs 4
Exp 6	dataset 4	dataset 1		4 vs 3

The datasets in Table V were assembled to simulate six cases for experimentation in heterogeneous domain adaptation as outlined in Table VII. The second and third columns indicate the origin of the source data and target data. The fourth column shows the dimensionality of the source data and target data, while the last columns show the number of fuzzy rules present in each domain.

2) *Regression Results*

The RMSE of the models on the target data $\boldsymbol{H}_U$ for the six experiments are shown in Table VIII.

TABLE VIII
RESULTS OF THE EXPERIMENTS IN HETEROGENEOUS DOMAIN ADAPTATION

Source to target datasets	RMSE of the models				
	$\overline{M}_s$	$\overline{M}_{t1}$	$\overline{M}_{t2}$	$\overline{M}_{t3}$	$\overline{M}_t$
Exp 1	9.5862± 0.0002	3.3168± 0.0021	2.9422± 0.0359	3.1510± 0.2120	2.9422± 0.0359
Exp 2	9.8785± 0.0003	7.0799± 0.0375	5.0758± 0.4040	4.7611± 0.0445	4.7611± 0.0445
Exp 3	8.2935± 0.0008	3.9622± 0.0333	3.2467± 0.1174	2.6774± 0.0090	2.6774± 0.0090
Exp 4	9.2541± 0.0009	2.9019± 0.3178	2.9033± 0.2476	2.5138± 1.0901	2.5138± 1.0901
Exp 5	9.6183± 0.0002	8.8253± 18.5659	3.9799± 1.8011	4.6508± 1.8893	3.9799± 1.8011
Exp 6	9.6353± 0.0001	2.6874± 0.1060	2.5636± 0.0141	2.5900± 0.0040	2.5636± 0.0141

From the results in Table VIII, we can conclude that the models $\overline{M}_{t1}$, $\overline{M}_{t2}$, and $\overline{M}_{t3}$ are all superior to the existing source model $\overline{M}_s$. In the six experiments, model $\overline{M}_{t2}$ showed the best performance in half the experiments, with model $\overline{M}_{t3}$ performing the best in the other half. Although model $\overline{M}_{t1}$ did not surpass the other two models, we intend to retain it as an alternative model. As mentioned in Section V, A2, the three models show vast differences in performance on different datasets, and the availability of different options for modifying an existing model enhance the probability of our method to successfully fit the data to the target domain.

TABLE IX
USING/NOT USING $\boldsymbol{H}_U$ – COMPARATIVE EXPERIMENTS

	RMSE of the models					
	$\overline{M}_{t1}$	$\overline{M}_{t1}$ (W)	$\overline{M}_{t2}$	$\overline{M}_{t2}$(W)	$\overline{M}_{t3}$	$\overline{M}_{t3}$(W)
Exp 1	3.3168± 0.0021	**3.3153± 0.0020**	2.9422± 0.0359	2.9487± 0.0380	3.1510± 0.2120	**3.1020± 0.1001**
Exp 2	7.0799± 0.0375	7.0860± 0.0405	5.0758± 0.4040	5.1412± 0.4184	4.7611± 0.0445	4.9762± 0.0084
Exp 3	3.9622± 0.0333	**3.9518±** 0.0121	3.2467± 0.1174	**3.2265±** 0.1688	2.6774± 0.0090	2.7944± 0.0505
Exp 4	2.9019± 0.3178	3.2590± 0.5608	2.9033± 0.2476	3.0016± 0.2099	2.5138± 1.0901	2.9480± 0.5821
Exp 5	8.8253± 18.5659	**8.6197±** 16.5161	3.9799± 1.8011	**3.4271±** 0.3989	4.6508± 1.8893	**4.1819±** 0.1494
Exp 6	2.6874± 0.1060	**2.5686±** 0.0532	2.5636± 0.0141	2.5778± 0.0167	2.5900± 0.0040	2.7160± 0.0332

Further, to verify the claim in Section IV, B that using target data without responses is risky, we conducted comparative experiments and provide those results in Table IX. The results in columns with "W" represent models that were constructed with the help of target data $\boldsymbol{H}_U$. These results indicate that using target data without responses has a negative function in model construction, especially in the methods that change the output space and change both the input and output space. Lower values are shown in bold.

3) *Parameter Sensitivity Analysis*

In the FHeDA method, the number of fuzzy rules used to construct the model in the latent feature space for the source domain is a significant parameter, because this also determines the number of fuzzy rules for the target domain. We also conducted the six experiments described in the last subsection with a varying $\bar{c}$. Table X shows the impact of $\bar{c}$ on model $\overline{M}_{t1}$, as an example.

TABLE X
RESULTS OF THE SENSITIVITY ANALYSIS FOR $\bar{c}$

$\bar{c}$	RMSE of the experiments					
	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	Exp 6
4	3.3001± 0.0110	6.9353± 0.0141	3.1835± 0.0019	3.1793± 0.1117	5.0963± 1.6004	2.8555± 0.0290
5	3.2807± 0.0079	6.9348± 0.0130	3.1703± 0.0022	3.7702± 3.1078	5.8903± 5.2418	2.7982± 0.0124
6	3.2774± 0.0064	6.9372± 0.0134	3.1649± 0.0193	3.0173± 0.0524	5.3352± 5.4235	2.8711± 0.0438
7	3.2842± 0.0090	6.9370± 0.0129	3.1738± 0.0252	4.2752± 6.1490	4.5622± 0.3267	2.7861± 0.0172
8	3.2806± 0.0084	6.9346± 0.0129	3.2164± 0.0030	3.7475± 3.3048	5.5040± 8.9666	2.8241± 0.0095

From the results, we can see that model $\overline{M}_{t1}$'s performance was slightly different when $\bar{c}$ was assigned with different values. The RMSE variance as $\bar{c}$ changes is small in "Exp 1, 2, 3, and 6". But when $\bar{c}$ is assigned with 5, 7, and 8, the RMSE variances are large in a few cases in "Exp 4" and in almost all cases in "Exp 5". We attribute this to too little target data with responses – so little data with responses, there isn't enough to represent the characteristics of the entire target dataset. The results in Table VII for "Exp 5" also verify this.

B. *Real-world Datasets*

Three real-world datasets from the UCI Machine Learning Repository were used to validate the FHeDA method in heterogeneous domain adaptation. Like the experiments in the homogeneous domain adaptation, the datasets from the machine leaning area are modified to simulate the scenarios of heterogeneous domain adaptation problems. The detailed description of these datasets are given.

The "Concrete Compressive Strength" dataset aims to predict the concrete compressive strength based on eight features, such as cement content, blast furnace slag, and fly ash. This dataset is dedicated to general regression tasks, so it needed to be revised in several respects to simulate a heterogeneous domain adaptation problem. First, the dataset was split into a source domain and a target domain using the "age" feature; instances with an age of less than 100 were treated as data in the target domain, the remainder fell into the source domain. To exacerbate the gap between the source and target domains, the features “blast furnace slag”, “fly ash” and “superplasticizer” were perturbed with random numbers following the normal distributions $N(0.1, 0.1)$ and $N(5,1)$ for source data and target data, respectively. The feature “age” in the source domain was then removed creating heterogeneous spaces across the two domains. Ultimately, we arrived at two datasets: one 7-dimensional source domain containing 110 data with responses, and one 8-dimensional target domain including 30 with responses and 80 without responses.

In the “Istanbul stock exchange”, two attributes “stock exchange returns” and the “Istanbul stock exchange national 100 index” were used to predict the “MSCI emerging marks index”. The first 200 instances fell into the source domain, and the last 200 instances were used as the target data. The two features were perturbed with random numbers following normal distribution $N(0.1, 0.1)$ for the source data and $N(5,1)$ for the target data. Further, the first feature in the source domain was discarded. Again, we arrived at the two datasets: one 1-dimensional source domain containing 200 data with responses, and one 2-dimensional target domain including 30 with responses and 170 data without responses.

In the last dataset, “air quality”, the features “temperature” and “relative humidity” were chosen as the input data, and absolute humidity“ was chosen as the output. The dataset was split based on the “relative humidity” value. Data with a “relative humidity” of greater than 0.5 formed the source domain, the remaining data was used for the target domain. The second feature in the target domain was discarded. Both features in the source domain were perturbed with random numbers following the normal distributions $N(0.1, 0.1)$ and $N(7,1)$, respectively. The input data of the target domain was changed with random numbers following normal distributions $N(0.1, 0.1)$. The final two datasets were: one 2-dimensional source domain containing 1200 data with responses, and one 1-dimensional target domain including 30 with responses and 1170 without responses.

To further prove our assertion that a model will not perform as well when trained only using a small amount of target data with responses in heterogeneous domain adaptation settings, we constructed and compared four models. The first was built using source data in a latent feature space $\overline{M}_s$. The second was constructed using insufficient target data with responses in the original feature space $\widetilde{M}_{t1}$. The third was built using target data with responses in a latent feature space $\widetilde{M}_{t2}$. And the fourth was built using the proposed FHeDA method.

The experimental results of these three real-world datasets are shown in Table XI.

From the results presented in Table XI, we can see that the performance of models $\widetilde{M}_{t1}$ and $\widetilde{M}_{t2}$ on the target data was poor, confirming our assumption that a model does not perform as well when being trained on less data. The target model $\overline{M}_t$ built using our method was superior to both the existing source model, and the model constructed using only a small amount of target data with responses.

TABLE XI
THE EXPERIMENTAL RESULTS FOR FHEDA ON REAL-WORLD DATA

Datasets	RMSE of the models			
	$\overline{M}_s$	$\widetilde{M}_{t1}$	$\widetilde{M}_{t2}$	$\overline{M}_t$
Concrete compressive strength	74.7001± 3968.0679	13144.4558± 8.0011e+08	17.8540± 36.3314	2.0976± 0.8587
Istanbul stock exchange	0.1749± 0.0000	102.9022± 7314.5177	1352.4233± 2.0321e+06	0.1409± 0.0000
Air quality	0.1501± 0.0000	0.1401± 0.0000	186.0764± 1.7286e+05	0.1365± 0.0000

VII. CONCLUSIONS AND FUTURE WORK

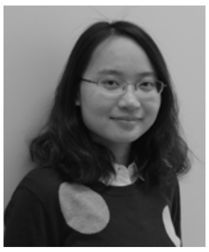
This study presented two methods for solving regression problems using fuzzy rule-based models in situations that require domain adaptation. The FHoDA method handles homogeneous spaces, and the FHeDA method handles heterogeneous spaces. In homogeneous domain adaptation, FHoDA solves mismatching fuzzy rules between the source and target domains and searches the h-nearest target data without responses around each target data with responses to improve the target model’s performance. In heterogeneous domain adaptation, a latent feature space is extracted to minimize the gap between the feature spaces of the two domains. Heterogeneous domain adaptation is converted into homogeneous domain adaptation after mapping all the input data from both domains into the new latent feature space. Experiments completed on synthetic and real-world datasets verify that the proposed methods greatly improve the performance of existing models when solving regression tasks in the target domain in both homogeneous and heterogeneous domain adaptation settings.

The presented methods offer three avenues for modifying the existing source model: changing the input space, changing the output space, or changing both. The model with the best performance is subsequently selected as the target model. Future studies will explore an algorithm that can recognize the differences between the fuzzy rules in the two domains in advance, so we can intentionally adopt a specific algorithm to modify an existing model.

REFERENCES

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering,* vol. 22, no. 10, pp. 1345-1359, 2010.
[2] M. Xiao and Y. Guo, "Feature space independent semi-supervised domain adaptation via kernel matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 37, no. 1, pp. 54-66, 2015.
[3] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong, "Transductive multi-view zero-shot learning," *IEEE transactions on pattern analysis and machine intelligence,* vol. 37, no. 11, pp. 2332-2345, 2015.
[4] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks,* vol. 22, no. 2, pp. 199-210, 2011.
[5] T. Tommasi, F. Orabona, and B. Caputo, "Learning categories from few examples with multi model knowledge transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 36, no. 5, pp. 928-941, 2014.
[6] J. Xu, S. Ramos, D. Vázquez, and A. M. Lopez, "Domain adaptation of deformable part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 36, no. 12, pp. 2367-2380, 2014.

[7] M. Chen, Z. Xu, K. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," *arXiv preprint arXiv:1206.4683,* 2012.
[8] M. Long, J. Wang, Y. Cao, J. Sun, and S. Y. Philip, "Deep Learning of transferable representation for scalable Domain Adaptation," *IEEE Transactions on Knowledge and Data Engineering,* vol. 28, no. 8, pp. 2027-2040, 2016.
[9] M. Gönen and A. A. Margolin, "Kernelized Bayesian Transfer Learning," in *AAAI*, 2014, pp. 1831-1839.
[10] D. Oyen and T. Lane, "Bayesian discovery of multiple Bayesian networks via transfer learning," in *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, 2013, pp. 577-586: IEEE.
[11] R. A. Bianchi, L. A. Celiberto, P. E. Santos, J. P. Matsuura, and R. L. de Mantaras, "Transferring knowledge as heuristics in reinforcement learning: A case-based approach," *Artificial Intelligence,* vol. 226, pp. 102-121, 2015.
[12] M. Klenk and K. Forbus, "Analogical model formulation for transfer learning in AP Physics," *Artificial Intelligence,* vol. 173, no. 18, pp. 1615-1638, 2009.
[13] D. Cook, K. D. Feuz, and N. C. Krishnan, "Transfer learning for activity recognition: A survey," *Knowledge and Information Systems,* vol. 36, no. 3, pp. 537-556, 2013.
[14] 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.
[15] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 26, no. 5, pp. 1019-1034, 2015.
[16] B. Gong, K. Grauman, and F. Sha, "Learning kernels for unsupervised domain adaptation with applications to visual object recognition," *International Journal of Computer Vision,* vol. 109, no. 1-2, pp. 3-27, 2014.
[17] Y. Shi and F. Sha, "Information-theoretical learning of discriminative clusters for unsupervised domain adaptation," *arXiv preprint arXiv:1206.6438,* 2012.
[18] R. Gopalan, R. Li, and R. Chellappa, "Unsupervised adaptation across domain shifts by generating intermediate data representations," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 36, no. 11, pp. 2288-2302, 2014.
[19] W. M. Kouw, L. J. Van Der Maaten, J. H. Krijthe, and M. Loog, "Feature-level domain adaptation," *Journal of Machine Learning Research,* vol. 17, no. 171, pp. 1-32, 2016.
[20] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 39, no. 7, pp. 1414-1430, 2017.
[21] X. Shi, Q. Liu, W. Fan, and S. Y. Philip, "Transfer across completely different feature spaces via spectral embedding," *IEEE Transactions on Knowledge and Data Engineering,* vol. 25, no. 4, pp. 906-918, 2013.
[22] C. Wang and S. Mahadevan, "Heterogeneous domain adaptation using manifold alignment," in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, 2011, vol. 22, no. 1, p. 1541.
[23] H. V. Nguyen, H. T. Ho, V. M. Patel, and R. Chellappa, "DASH-N: Joint hierarchical domain adaptation and feature learning," *IEEE Transactions on Image Processing,* vol. 24, no. 12, pp. 5479-5491, 2015.
[24] Y.-R. Yeh, C.-H. Huang, and Y.-C. F. Wang, "Heterogeneous domain adaptation and classification by exploiting the correlation subspace," *IEEE Transactions on Image Processing,* vol. 23, no. 5, pp. 2009-2018, 2014.
[25] 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, no. 2, pp. 1637-1646, 2014.
[26] V. Behbood, J. Lu, G. Zhang, and W. Pedrycz, "Multistep fuzzy bridged refinement domain adaptation algorithm and its application to bank failure prediction," *IEEE Transactions on Fuzzy Systems,* vol. 23, no. 6, pp. 1917-1935, 2015.
[27] 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, no. 4, pp. 597-609, 2013.
[28] Z. Deng, K.-S. Choi, Y. Jiang, and S. Wang, "Generalized hidden-mapping ridge regression, knowledge-leveraged inductive transfer learning for neural networks, fuzzy systems and kernel methods," *IEEE transactions on cybernetics,* vol. 44, no. 12, pp. 2585-2599, 2014.
[29] Z. Deng, Y. Jiang, L. Cao, and S. Wang, "Knowledge-leverage based TSK fuzzy system with improved knowledge transfer," in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE),*, 2014, pp. 178-185.
[30] C. Yang, Z. Deng, K.-S. Choi, and S. Wang, "Takagi–Sugeno–Kang transfer learning fuzzy logic system for the adaptive recognition of epileptic electroencephalogram signals," *IEEE Transactions on Fuzzy Systems,* vol. 24, no. 5, pp. 1079-1094, 2016.
[31] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Fuzzy Regression Transfer Learning in Takagi–Sugeno Fuzzy Models," *IEEE Transactions on Fuzzy Systems,* vol. 25, no. 6, pp. 1795-1807, 2017.
[32] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Granular Fuzzy Regression Domain Adaptation in Takagi-Sugeno Fuzzy Models," *IEEE Transactions on Fuzzy Systems,* vol. 26, no. 2, pp. 847-858, 2017.
[33] F. Liu, G. Zhang, H. Lu, and J. Lu, "Heterogeneous Unsupervised Cross-domain Transfer Learning," *arXiv preprint arXiv:1701.02511,* 2017.
[34] M. L. Hadjili and V. Wertz, "Takagi-Sugeno fuzzy modeling incorporating input variables selection," *IEEE Transactions on fuzzy systems,* vol. 10, no. 6, pp. 728-742, 2002.
[35] Y. Jiang, F.-L. Chung, H. Ishibuchi, Z. Deng, and S. Wang, "Multitask TSK fuzzy system modeling by mining intertask common hidden structure," *IEEE Transactions on Cybernetics,* vol. 45, no. 3, pp. 534-547, 2015.
[36] Y. Jiang, Z. Deng, F.-L. Chung, and S. Wang, "Realizing two-view TSK fuzzy classification system by using collaborative learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 47, no. 1, pp. 145-160, 2017.
[37] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Computers & Geosciences,* vol. 10, no. 2-3, pp. 191-203, 1984.



Hua Zuo is a postdoctoral research associate with the Faculty of Engineering and Information Technology, University of Technology Sydney, Australia.

Her research interests include transfer learning and fuzzy systems.

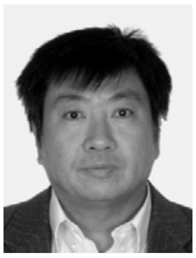
She is a Member of the Decision Systems and e-Service Intelligence (DeSI) Research Laboratory at the Centre for Artificial Intelligence, University of Technology Sydney.

Jie Lu (SM'13) is a Distinguished Professor, the Director of the Centre for Artificial Intelligence, and the Associate Dean (Research Excellence) with the Faculty of Engineering and Information Technology at the University of Technology Sydney, Australia. She received her PhD in information systems from the Curtin University of Technology, Australia, in 2000.

Her research expertise spans fuzzy transfer learning, decision support systems, recommender systems, 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 170 papers in IEEE Transactions and other international journals. She has been awarded eight Australian Research Council (ARC) Discovery Project grants and many other research grants. She is a member of the ARC College of Experts.

She serves as Editor-In-Chief for *Knowledge-Based Systems* (Elsevier), Editor-In-Chief for the *International Journal on*

Computational Intelligence Systems (Atlantis), Associate Editor for *IEEE Transactions on Fuzzy Systems*, Editor for a book series on Intelligent Information Systems (World Scientific), and has served as a guest editor of 12 special issues, general/PC/organization chairs for ten international conferences as well as having delivered 16 keynote/plenary speeches at IEEE and other international conferences.



Guangquan Zhang is an Associate Professor and the Director of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory at the Center for Artificial Intelligence, Faculty of Engineering and Information Technology, University of Technology Sydney, Australia. He received his PhD in applied mathematics from Curtin University of Technology, 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 seven Australian Research Council (ARC) Discovery Projects grants and many other research grants. He was awarded an 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 of IEEE Transactions and other international journals, and has co-chaired several international conferences and workshops in the area of fuzzy decision-making and knowledge engineering.

Witold Pedrycz (F'98) is a Professor and the Canada Research Chair (CRC) in Computational Intelligence of the Department of Electrical and Computer Engineering, University of Alberta, Canada. He received a PhD and DSci from the Silesian University of Technology, Poland. He is a foreign member of the Polish Academy of Sciences and 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 the 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***.