Granular Fuzzy Regression Domain Adaptation in Takagi-Sugeno Fuzzy Models

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

***Abstract*—In classical data-driven machine learning methods, massive amounts of labeled data are required to build a high-performance prediction model. However, the amount of labeled data in many real-world applications is insufficient, so establishing a prediction model is impossible. Transfer learning has recently emerged as a solution to this problem. It exploits the knowledge accumulated in auxiliary domains to help construct prediction models in a target domain with inadequate training data. Most existing transfer learning methods solve classification tasks; only a few are devoted to regression problems. In addition, the current methods ignore the inherent phenomenon of information granularity in transfer learning. In this study, granular computing techniques are applied to transfer learning. Three granular fuzzy regression domain adaptation methods to determine the estimated values for a regression target are proposed to address three challenging cases in domain adaptation. The proposed granular fuzzy regression domain adaptation methods change the input and/or output space of the source domain's model using space transformation, so that the fuzzy rules are more compatible with the target data. Experiments on synthetic and real-world datasets validate the effectiveness of the proposed methods.**

***Index Terms*—Machine learning, transfer learning, fuzzy rules, granular computing, regression**

I. Introduction

TRADITIONAL machine learning methods use learning models to extract knowledge from massive amounts of labeled data. They work under a common assumption that the training data (in the source domain) and the testing data (in the target domain) have the same feature space and the same probability distributions. However, if the feature space or the distribution of the target data changes, the models built from the source data become unsuitable and a new model needs to be rebuilt and trained from scratch. Additionally, if there is insufficient labeled target data, a new prediction model for the target data will be impossible to establish.

In recent years, new machine-learning method, called transfer learning, has been introduced as a solution to the above problem. Transfer learning leverages previously acquired knowledge in a similar domain to improve the efficiency and accuracy of learning in a target domain [1]. Web document classification is a well-known example of where transfer learning has been used to address a real-world problem – web documents are classified into predefined categories based on previously collected data [2, 3].

Many approaches to transfer learning have been proposed, and the related work can be divided into four categories according to the problem setting: multi-task learning [4], domain adaption [5], cross-domain adaptation [6], and heterogeneous learning [7]. Computational intelligence has recently been applied to improve the performance of existing transfer learning methods and to handle knowledge transfer processes in real-world systems. These computational intelligence techniques can be divided into three main categories [8]: neural network-based transfer learning [9], Bayesian transfer learning [10], and fuzzy transfer learning [11]. These techniques have a wide range of applications, including natural language processing [12], computer vision and image processing [13, 14], and biology [15].

A significant amount of transfer learning research has been undertaken for classification problems, yet studies on regression problems are still scarce. In this paper, we focus on regression problems using regression domain adaptation techniques.

Imprecision, approximation, vagueness, and ambiguity of information are driven by the variability encountered when trying to learn an activity with little information [16]. There is a clear co-dependency on the level of certainty in learning an activity and the amount of information that is available; problems with too little information have a high degree of uncertainty. For this reason, very recent studies have applied fuzzy techniques to transfer learning [17]. Using fuzzy logic allows for more approximation and greater uncertainty within the knowledge transfer. Behbood et al. [18] 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

Manuscript received Sep. 29, 2016; accepted Mar. 30, 2017. 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, Centre for Artificial Intelligence, 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).

during the refinement process. Using fuzzy techniques in the similarity measurement, the authors revealed the advantage of fuzzy logic in knowledge transfer when the target domain lacks critical information, is vague and involves uncertainty [19, 20]. Shell and Coupland [21, 22] introduced a novel framework for transfer learning, based on a fuzzy inference system, to address cases where only limited unlabeled target data and labeled source data are available. They used fuzzy rules from the source domain and adapted them to predict labels in the target domain, based on Euclidean distance measurements. Deng et al. [23, 24] improved knowledge leverage in Takagi-Sugeno-Kang (TSK) fuzzy models for inductive transfer learning. Two knowledge leverage strategies were proposed to boost learning of the antecedent and consequent parameters, and further training data in the target domain was applied to help learn the antecedents. Given that fuzzy system modeling is an important category of modeling with extensive applications [25, 26], and is an technique that performs well when addressing transfer learning problems, incorporating regression domain adaptation into a fuzzy model holds promise.

Although many approaches have been introduced as possible solutions for transfer learning problems, their performance is not yet acceptable. One reason is the information granularity inherent in many problems. For instance, 128GB of mobile storage is considered large today, whereas 32GB was regarded as large five years ago. The precise values, 128GB and 32GB, both need to be expressed as a granular value, "large", for learning to be effectively transferred from the 5-year-old domain to today's domain. Extracting additional abstract knowledge shared between domains should therefore assist knowledge transfer. Granular computing (GrC) is an emerging information processing paradigm that transforms complex data into information granules at different levels of resolution to reveal different features and irregularities. GrC's ability to address information at different levels of abstraction could improve the performance of transfer learning, and consequently we propose several granular fuzzy regression domain adaptation methods, GFRDA for short, to address regression domain adaptation problems.

In the previous paper [27], we proposed a method of changing the input space that is specific for the domain adaptation problem where the conditions of fuzzy rules in two domains are different. In the current paper, forming a natural follow-up of the previous paper, we proposed two methods other to deal with different cases in domain adaptation, namely changing the output space and changing both the input and output spaces. These three methods are based on Takagi-Sugeno fuzzy models, and each is designed to handle a different domain adaptation case: where conditions differ, conclusions differ, or both differ between the source and target domains. Furthermore, these three methods considered together constitute an overall framework that provides a comprehensive framework for the domain adaptation-based of fuzzy models.

The main contributions of this paper are twofold. First, the information granularity inherent in transfer learning is considered to effectively improve model performance for the target domain. Second, an entire framework is proposed to provide guidance for domain adaptation with fuzzy models.

The paper is structured as follows. Section II provides some definitions related to domain adaptation and GrC, and Takagi-Sugeno fuzzy models are introduced. Section III defines fuzzy domain adaptation and describes the three specific domain adaptation cases using fuzzy rule-based models. Section IV presents the corresponding GFRDA methods to handle these cases. The results of the experiments on synthetic and real-world datasets to analyze and verify the proposed GFRDA methods are presented in Sections V and VI. The final section concludes the paper and outlines future work.

II. Preliminaries

The definition of transfer learning, and particularly domain adaptation, is introduced in this section, followed by the Takagi-Sugeno fuzzy model and finally some GrC-related knowledge is described.

A. Definitions

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

Definition 2 (Task) [1]: A task is denoted by $T = \{Y, f(\cdot)\}$, where $Y \in R$ is an output value, 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 $T_s \neq T_t$.

Definition 4 (Domain Adaptation) [1]: Domain adaptation is a category of inductive transfer learning in which $F_t = F_s$, but $P_t(X) \neq P_s(X)$.

B. Takagi-Sugeno Fuzzy Models

The Takagi-Sugeno fuzzy model is an effective way to represent a fuzzy model in a nonlinear dynamic system. A Takagi-Sugeno model, composed of c fuzzy rules, is formally represented as:

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

Each fuzzy rule comprises one condition, which is described by the prototype $\boldsymbol{v}_i$, and one conclusion, which is typically governed by the coefficients of the linear function L_i of the input variables $\boldsymbol{a}_i$. When the input of the Takagi-Sugeno fuzzy model is $\boldsymbol{x}$, the output y is represented as:

$$y = \sum_{i=1}^{c} A_i(\boldsymbol{x}, \boldsymbol{v}_i)\, L_i(\boldsymbol{x}, \boldsymbol{a}_i) \tag{2}$$

The construction of this fuzzy rule-based model uses $\boldsymbol{S} = \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)\}$ to formulate condition A_i and optimize the parameters of L_i. The design procedure can be summarized in two steps [28]:

Step 1: Form the conditions $A_1, \dots, A_c$ through fuzzy clustering.

Typically, a Fuzzy C-Means (FCM) is used to construct the clusters and calculate the prototypes $\boldsymbol{v}_i$. An FCM partitions N data $\boldsymbol{x}_1, \dots, \boldsymbol{x}_N$ into c clusters, where $1 < c < N$. As a result, a collection of c prototypes $\boldsymbol{v}_1, \dots, \boldsymbol{v}_c$ is formed and the membership of instance $\boldsymbol{x}$ belonging to each cluster is calculated in the form:

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

where m $(m > 1)$ is a fuzzification coefficient that affects both the shape and overlap of the resulting membership functions.
Step 2: Optimize the parameters of the linear functions $L_i(\boldsymbol{x}, \boldsymbol{a}_i)$.

Given an input, the output of a Takagi-Sugeno fuzzy model is parameter $\boldsymbol{a}$'s linear function, so the optimal parameter $\boldsymbol{a}$ can be calculated using the given dataset $\boldsymbol{S}$ and represented as:

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

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 $\boldsymbol{y} = [y_1 \ \cdots \ y_N]^{\boldsymbol{T}}$.

Therefore, the fuzzy rule's conclusion can be calculated based on the derived $\boldsymbol{a}$:

$$L_i(\boldsymbol{x}, \boldsymbol{a}_i) = a_{i0} + a_{i1}x_1 + \cdots + a_{in}x_n \qquad i = 1, \dots, c \tag{5}$$

where $\boldsymbol{x} = [x_1 \ \cdots \ x_n]^T$, $\boldsymbol{a}_i = [a_{i0} \ a_{i1} \ \cdots a_{in}]$.

C. Granular Computing

Granular computing is an emerging information processing paradigm that transforms complex data into information granules at different solution levels. Information granules can be perceived as a collection of elements drawn together by their closeness (resemblance, proximity, functionality, etc.) and articulated in terms of useful spatial, temporal, or functional relationships. Granular computing (GrC) represents, constructs, and processes information granules.

Information granules are formalized in many different ways. Depending on the problem, different formalisms to represent the information granules have been applied, such as interval sets, fuzzy sets, rough sets, and shadowed sets. The level of granularity determines the level of detail used to classify the data. Different types of knowledge can be captured or learned by representing data with information granules at different levels [29]. Features and regularities in the data can emerge, while the detail is deliberately hidden [30]. For example, interesting cloud patterns representing a cyclone may be noticable in a low-resolution satellite image, while in a higher-resolution image, this large-scale atmospheric phenomenon might be missed. High resolution images are more useful for observing small-scale phenomenon, such as an interesting street pattern in Manhattan.

The most significant perspective we introduce with regard to GrC is that it is possible to obtain different levels of knowledge when dealing with data represented by information granules that have different levels of granularity. The higher the level of granularity, the more abstract the knowledge obtained.

III. Knowledge Transfer in Takagi-Sugeno Fuzzy Models

This section consists of two subsections. The first subsection defines fuzzy domain adaptation, as distinct from the original definition of domain adaptation. The second subsection describes domain adaptation in fuzzy rule-based systems, analyzes transfer learning problems from the perspective of GrC, and proposes methods that address three cases that occur fuzzy domain adaptation.

A. Definitions of Fuzzy Domain Adaptation

In the original definition of transfer learning, the source domain and the target domain are distinguished by the feature space, the probability distribution in the domain, and by the task, usually represented by a prediction function. In domain adaptation, which is a category of inductive transfer learning, the source domain and the target domain have the same feature space but a different distribution [1]. Most works in the computational intelligence area on domain adaptation are based on this definition and apply neural or Bayes networks as the basic learning model.

We use a Takagi-Sugeno fuzzy model as the basic model for our learning tasks. Since the characteristics of this fuzzy rule-based model are not the same as a neural or Bayesian networks, domain adaptation must be redefined for fuzzy systems as follows:
Definition 5 (Fuzzy Domain Adaptation): In a Takagi-Sugeno fuzzy model, a source domain and a target domain are represented as:
Source domain: $D_s = \{F_s = (F_1, \cdots, F_n),\ G_s(\boldsymbol{x}) = (G_{s1}(\boldsymbol{x}), \cdots, G_{sc}(\boldsymbol{x})),\ L_s = \{L_{s1}, \cdots, L_{sc}\}\}$
Target domain: $D_t = \{F_t = (F_1, \cdots, F_n),\ G_t(\boldsymbol{x}) = (G_{t1}(\boldsymbol{x}), \cdots, G_{tc}(\boldsymbol{x})),\ L_t = \{L_{t1}, \cdots, L_{tc}\}\}$
where F_s and F_t are the feature spaces in two domains. $G_{s1}, \cdots, G_{sc}$ and $G_{t1}, \cdots, G_{tc}$ are the constructed fuzzy sets in two domains, and $G_{s1}(\boldsymbol{x}), \cdots, G_{sc}(\boldsymbol{x})$ and $G_{t1}(\boldsymbol{x}), \cdots, G_{tc}(\boldsymbol{x})$ form the membership functions, which determine the condition parts of the fuzzy rules. l_s and l_t are the linear functions that govern the conclusion parts of the fuzzy rules. $F_s = F_t$, $G_s(\boldsymbol{x}) \neq G_t(\boldsymbol{x})$, and/or $L_s \neq L_t$.

In fuzzy domain adaptation, the feature spaces in the two domains are the same, $F_s = F_t$, but either the fuzzy sets or the linear functions or both are different across the two domains, giving $G_s(\boldsymbol{x}) \neq G_t(\boldsymbol{x})$, and/or $L_s \neq L_t$. In general, we consider that $G_s(\boldsymbol{x}) = G_t(\boldsymbol{x})$ means $G_{si}(\boldsymbol{x}) = G_{ti}(\boldsymbol{x})$, $i = 1, \cdots, c$, and $G_{si}(\boldsymbol{x}) \neq G_{ti}(\boldsymbol{x})$ means $G_{si}(\boldsymbol{x}) \neq G_{ti}(\boldsymbol{x})$, $i = 1, \cdots, c$. Similarly, $L_s = L_t$ indicates $L_{si} = L_{ti}$, $i = 1, \cdots, c$, and $L_s \neq L_t$ indicates $L_{si} \neq L_{ti}$, $i = 1, \cdots, c$.

B. Knowledge Transfer Learning in Granular Fuzzy Models

According to the fuzzy domain adaptation model defined above, the discrepancies between the source domain and the target domain can be summarized according to one of three cases: having different conditions, different conclusions, or both. To emphasize the difference, the source domain's model is fixed, and the target domain's model is varied according to these three cases.

Suppose fuzzy model M in the source domain, described in the form of the fuzzy rules, is:

Model M:

$$\text{if } \boldsymbol{x} \text{ is } A_i(\boldsymbol{x}, \boldsymbol{v}_i), \text{ then } y \text{ is } L_i(\boldsymbol{x}, \boldsymbol{a}_i) \qquad i = 1, \ldots, c \tag{6}$$

The fuzzy models in the target domain that correspond to the three cases in fuzzy domain adaptation, described in the form of the fuzzy rules, are:

Model $\widetilde{M}_1$:

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

Model $\widetilde{M}_2$:

$$\text{if } \boldsymbol{x}' \text{ is } A_i(\boldsymbol{x}', \boldsymbol{v}_i), \text{ then } y' \text{ is } L_i(\boldsymbol{x}', \boldsymbol{a}_i') \qquad i = 1, \ldots, c \tag{8}$$

Model $\widetilde{M}_3$:

$$\text{if } \boldsymbol{x}' \text{ is } A_i(\boldsymbol{x}', \boldsymbol{v}_i'), \text{ then } y' \text{ is } L_i(\boldsymbol{x}', \boldsymbol{a}_i') \qquad i = 1, \ldots, c \tag{9}$$

In the first case, comparing models M and $\widetilde{M}_1$, the conditions of the fuzzy rules in the two domains are different, but the conclusions are the same. In the second case, comparing models M and $\widetilde{M}_2$, the conditions of the fuzzy rules are the same, but the conclusions are totally different. In the third case, comparing models M and $\widetilde{M}_3$, the conditions and conclusions of the fuzzy rules in both the source and target domains are different.

There are now massive amounts of labeled data in the source domain and a well-performing model M can be built. In the target domain, there is a large amount of unlabeled data and little labeled data, so establishing a prediction model is impossible. Because the fuzzy rules in the two domains are different, the model for the source domain M is not suited to regression tasks in the target domain.

Next, domain adaptation problems are analyzed from the perspective of GrC. The knowledge contained in both the source and target domains can be treated as information granules. Since the information granules in each domain have different levels of granularity, a model based solely on knowledge from the source domain could not directly solve tasks in the target domain. For example, RAM is an important index for predicting the price of a computer. Thirty years ago, computers typically had 256kb of RAM, whereas now 8G is fairly standard. These two values, 256kb and 8G, can both be treated as information granules, but with different granularity levels as their unit of measurement are different. Therefore, the knowledge gleaned from data based on 256kb is not suitable for tasks relevant to the 8G information.

The higher the granularity level in GrC, the more abstract the knowledge extracted. Based on the existing knowledge (information granules) in the source domain, our idea is to extract and construct information granules at a higher level of granularity so that knowledge can be appropriately shared between the two domains. However, the knowledge contained in the new information granules cannot be directly used to solve tasks in the target domain since the required level of granularity is different. An additional procedure is needed to transform the new information granules to a lower level, so they can be applied to help solve the target tasks. The essence of this process is shown in Fig. 1.

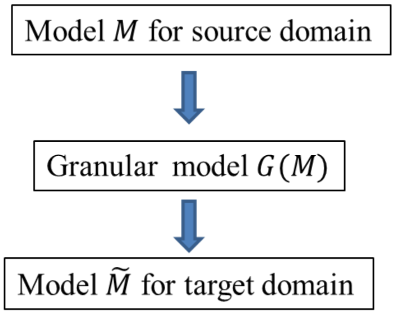
Model M for source domain
Granular model $G(M)$
Model $\widetilde{M}$ for target domain

Fig. 1. Knowledge transfer from a GrC perspective

The process has two steps. A granular model is built by transforming the information granules from a lower level to a higher level, and the granularity level of the new granules is reduced to suit the target domain.

Continuing the example of the computer's internal storage, our aim is to use a more abstract representation to describe a computer's RAM 30 years ago. For instance "large capacity" instead of a numerical value: 256kb. We can still say a computer of today has "large capacity" if its RAM meets or exceeds 8G. In this example, "large capacity" is treated as an information granule with a higher granularity level that builds a bridge to connect two granules of lower level.

Instead of conducting the two steps in Fig. 1 separately, we implement them simultaneously. Because the results of the first step significantly impact the performance of the following procedure, merging the two steps benefits the method's execution. A nonlinear space transformation is used to achieve these two steps, and an optimization process makes the resulting model more compatible with the tasks in the target domain.

Different strategies are applied in the three domain adaptation cases to implement the above process. Where the conditions of the fuzzy rules in two domains are different, the conditions are changed using a space transformation so that the transformed fuzzy rules approximate the expected model in the target domain. Where the conclusions of the fuzzy rules are different, the conclusions are changed using mapping to ensure the newly constructed model is as close as possible to the expected target model. Where both the conditions and the conclusions are different, a method that modifies both the conditions and conclusions is used so that the transformed fuzzy rules approximate the expected fuzzy rules and are more compatible with the target domain.

Since the conditions of fuzzy rules are governed by the input data, the method that changes the conditions can be regarded as transforming the input space. The conclusions determine the output of the fuzzy rules, so this method transforms the output space. Similarly, the last method transforms both the input and output spaces.

A simple example with one-dimensional input data is shown in Fig. 2 to illustrate space transformation in our prosed model.

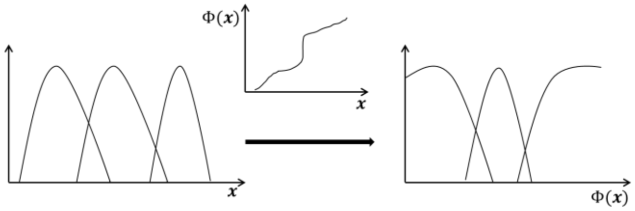

Fig. 2. An example of conditions of fuzzy rules under space transformation

The left section of Fig. 2 shows the membership functions of the fuzzy sets constructed with FCM. Using the space transformation Φ, the input variable $\boldsymbol{x}$ becomes $\Phi(\boldsymbol{x})$. More importantly, the membership functions in the new space have been changed, as shown on the right.

IV. The Three GRFDA Methods

This section describes the specific procedures of the proposed GFRDA methods, followed by the performance index used to evaluate the constructed models.

A. The Three GRFDA Methods

In this paper, the assumption is that there is a mass of labeled data in the source domain, and only a few labeled data and many unlabeled data in the target domain. Suppose the dataset in the source domain is $\boldsymbol{S}=\{(\boldsymbol{x}_1,y_1),\cdots,(\boldsymbol{x}_N,y_N)\}$, and the dataset in the target domain is $\boldsymbol{H}=\{\boldsymbol{H}_L,\boldsymbol{H}_U\}=\left\{(\boldsymbol{x}'_1,y'_1),\cdots,\left(\boldsymbol{x}'_{N'_t},y'_{N'_t}\right),\ \boldsymbol{x}'_{N'_t+1},\cdots,\boldsymbol{x}'_{N'}\right\}$, where the data in $\boldsymbol{H}_L$ has labels (called estimated values in regression problems), and data in $\boldsymbol{H}_U$ has no labels. The number of data in $\boldsymbol{S}$ is N, the total number of data in $\boldsymbol{H}$ is N', the number of data in $\boldsymbol{H}_L$ is N'_t, the number of data in $\boldsymbol{H}_U$ is $N'-N'_t$, and N'_t satisfies $N'_t \ll N$ and $N'_t \ll N'$.

In the proposed GFRDA methods, the process of transferring knowledge from the source domain to the target domain has two steps. First a Takagi-Sugeno fuzzy model based on source data is constructed; second, a new fuzzy model for the target domain is built by modifying the input and/or output space of the existing model (fuzzy rules). The first step is the same for all three methods, while the second step differs depending on the method. This process is shown in Fig. 3.

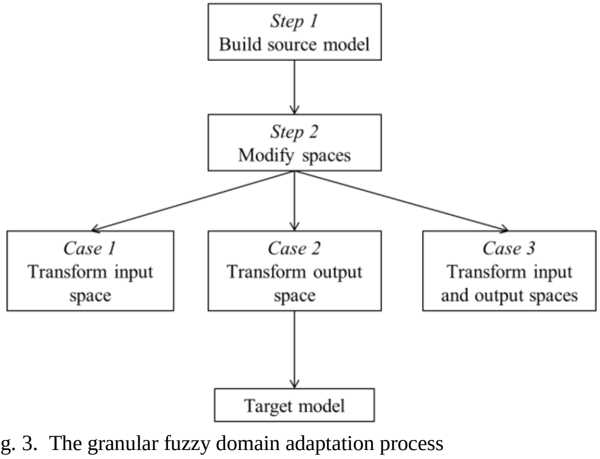

Fig. 3. The granular fuzzy domain adaptation process

The procedure for Step 1, building a fuzzy model for the source domain, follows.

Step 1: Construct a Takagi-Sugeno fuzzy model M based on source data.

A Takagi-Sugeno fuzzy model M is constructed using source data $\boldsymbol{S}$.

Model M

$$\text{if } \boldsymbol{x}_k \text{ is } A_i(\boldsymbol{x}_k,\boldsymbol{v}_i)\text{, then } y_k \text{ is } L_i(\boldsymbol{x}_k,\boldsymbol{a}_i) \qquad i=1,\dots,c \tag{10}$$

The main blocks of fuzzy rules are the conditions and conclusions, which are dominated by prototypes of the data and linear functions, respectively. Model M is therefore constructed by calculating the data prototypes and the linear functions through the procedures described in Section II.A. Thus, we have the prototypes $\boldsymbol{v}_1,\cdots,\boldsymbol{v}_c$, and the linear functions $L_1(\cdot,\boldsymbol{a}_1),\cdots,L_c(\cdot,\boldsymbol{a}_c)$.

We now take some data from the dataset $\boldsymbol{H}$ in the target domain; however, model M does not perform well on dataset $\boldsymbol{H}$, since these data follow a different fuzzy model and different fuzzy rules to those of model M. The number of labelled data in dataset $\boldsymbol{H}_L$ is not sufficiently large to build a good model for the target domain, so the proposed methods apply knowledge from the source domain to help the target domain build a new model.

In the second step, the input and/or output space of model M obtained in Step 1 is modified through mappings using the labeled target data $\boldsymbol{H}_L$ to build a new fuzzy model for the target domain.

Step 2: Modify the existing fuzzy rules to build a new fuzzy regression model for the target domain.

The three different domain adaptation cases, shown in (7) – (9), are considered, and the steps for the corresponding GFRDA method are explained below.

Step 2a) Method 1: change the input space

To handle the cases where the fuzzy rules' conditions in the source and target domains are not identical, we apply the method proposed in our previous paper [27]. The target domain's ideal model is described in (7). Since the way of constructing the space transformation is the same in these three methods, we detail it in this method and not repeat in the other two methods.

Since there is insufficient labeled data to train the fuzzy model $\widetilde{M}_1$, the learned knowledge (fuzzy rules) in the existing model M is used to help the target domain construct a new fuzzy model. In this method, the input space is changed by optimizing a continuous mapping for each input variable. Through mapping Φ, the input space is transformed to $\Phi(\boldsymbol{x}')$, and the new fuzzy model M'_1 for the target domain is constructed using the fuzzy rules from model M. This process and resulting architecture are shown in Fig. 4.

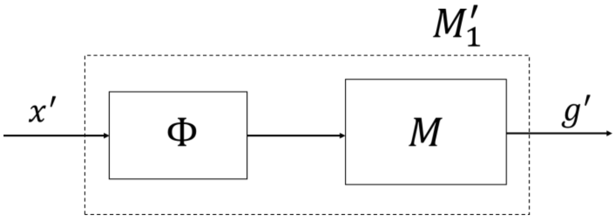

Fig. 4. Method 1: changing the input space

Model M_1', described in the form of fuzzy rules, is:

$$\text{if } \boldsymbol{x}' \text{ is } A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i)), \text{ then } g' \text{ is } L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i) \quad i = 1, \ldots, c \tag{11}$$

Because mapping Φ is a transformation of the input space, the changes are reflected in the input data and the prototypes (the centers of the clusters) with the forms $\Phi(\boldsymbol{x}')$ and $\Phi(\boldsymbol{v}_i)$.

Therefore the output of model M_1' is:

$$g' = \sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i)) \; L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i) \tag{12}$$

Our aim is to find a Φ such that $M_1' \approx \widetilde{M}_1$, i.e.,

$$\sum_{k=1}^{N_t'} g_k' \approx \sum_{k=1}^{N_t'} y_k' \tag{13}$$

i.e.

$$\sum_{k=1}^{N_t'} \sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i)) \, L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i) \approx \sum_{k=1}^{N_t'} y_k' \tag{14}$$

The parameters of Φ are optimized by minimizing the objective function as follows:

$$Q_1' = \sqrt{\frac{1}{N_t'} \sum_{k=1}^{N_t'} (\sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'), \Phi(\boldsymbol{v}_i) L_i(\Phi(\boldsymbol{x}'), \boldsymbol{a}_i) - y_k')^2} + \frac{\lambda}{2} w^T w \tag{15}$$

The first term in (15) is the approximation error that aims to minimize the gap between the output of model M_1' and the target data's real output. The second term introduces a structural risk term into the objective function. The parameter λ indicates the tradeoff between the quality of an approximation and the complexity of the approximation function; w is the vector of all the parameters optimized.

The mapping is the key element in each of our GFRDA methods. We use nonlinear continuous functions, composed of sigmoid functions, to construct the mappings of Φ. Fig. 5 shows the jth input variable of data $\boldsymbol{x}_k'$ as an example of the structure of the mapping for each input variable.

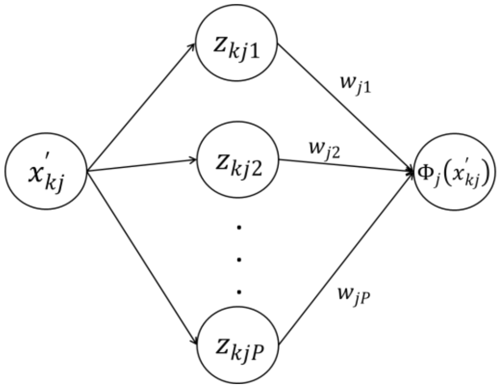

Fig. 5. Architecture of nonlinear mapping

The mapping is composed of P nodes in the hidden layer and a single node at the output layer that constructs the network. The two parametric sigmoid functions are applied to the hidden nodes, where the pth sigmoid function for the jth input variable of $\boldsymbol{x}_k'$ is:

$$z_{kjp} = \frac{1}{1+e^{-\alpha_{jp}(x_{kj}'-\beta_{jp})}} \tag{16}$$

$j = 1, \ldots, n$, $p = 1, \ldots, P$, $\alpha_{jp} > 0$.

Therefore, the transformation of input variable x_{kj}' under mapping Φ_j is:

$$\Phi_j(x_{kj}') = \sum_{p=1}^{P} w_{jp} * z_{kjp} \tag{17}$$

where w_{jp} represents the weight of the pth sigmoid function of the output variable, and satisfies $\sum_{p=1}^{P} w_{jp} = \max\{x_{1j}', \cdots, x_{N'j}'\}$. z_{kjp} is calculated through Eq. (16).

$\Phi = [\Phi_1 \; \Phi_2 \; \cdots \; \Phi_n]$, where Φ_j is constructed following the procedure described above. Thus the input data $\boldsymbol{x}_k'$ becomes $\Phi(\boldsymbol{x}_k')$:

$$\Phi(\boldsymbol{x}_k') = \begin{bmatrix} \Phi_1(x_{k1}') \\ \Phi_2(x_{k2}') \\ \cdots \\ \Phi_n(x_{kn}') \end{bmatrix} = \begin{bmatrix} \sum_{p=1}^{P} w_{1p} \; \frac{1}{1+e^{-\alpha_{1p}(x_{k1}'-\beta_{1p})}} \\ \sum_{p=1}^{P} w_{2p} \; \frac{1}{1+e^{-\alpha_{2p}(x_{k2}'-\beta_{2p})}} \\ \cdots \\ \sum_{p=1}^{P} w_{np} \; \frac{1}{1+e^{-\alpha_{np}(x_{kn}'-\beta_{np})}} \end{bmatrix} \tag{18}$$

Taking advantage of the nonlinear mappings, transformations are made to the input space so that the new input variables become more compatible with the data in the target domain. The parameters of Φ are derived through an optimization process by minimizing (17) using the labeled dataset $\boldsymbol{H}_L$. The dataset $\boldsymbol{H}_U$ is used to test the performance of the model after its construction.

Step 2b) Method 2: change the output space

This method handles cases where the conclusions of the fuzzy rules in the two domains are different. The target domain's ideal model is described in (8).

Again, there are sufficient data to train the fuzzy model $\widetilde{M}_2$

for the target domain. Since the conclusions are different to those in the source domain, we introduce a method to modify the output space by optimizing a continuous mapping for each output. The output space is modified by mapping Ψ, and a new fuzzy model M_2' for the target domain is constructed based on the fuzzy rules in model M. The process and resulting architecture is shown in Fig. 6.

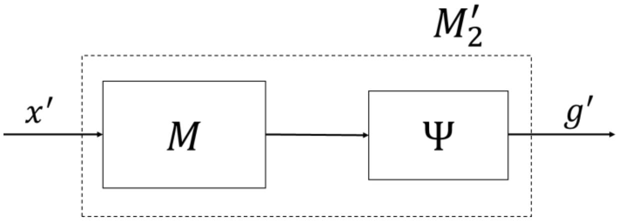

Fig. 6. Method 2: changing the output space

Model M_2', described in the form of fuzzy rules, is:

$$\text{if } \boldsymbol{x}' \text{ is } A_i(\boldsymbol{x}',\boldsymbol{v}_i), \text{, then } g' \text{ is } \Psi_i(L_i(\boldsymbol{x}',\boldsymbol{a}_i)) \quad i = 1,\dots,c \tag{19}$$

Therefore, the output of model M_2' is:

$$g' = \sum_{i=1}^{c} A_i(\boldsymbol{x}',\boldsymbol{v}_i)\ \Psi_i(L_i(\boldsymbol{x}',\boldsymbol{a}_i)) \tag{20}$$

Our aim is to find a Ψ such that $M_2' \approx \widetilde{M}_2$, i.e.,

$$\sum_{k=1}^{N_t'} g_k' \approx \sum_{k=1}^{N_t'} y_k' \tag{21}$$

i.e.

$$\sum_{k=1}^{N_t'}\sum_{i=1}^{c} A_i(\boldsymbol{x}',\boldsymbol{v}_i)\,\Psi_i(L_i(\boldsymbol{x}',\boldsymbol{a}_i)) \approx \sum_{k=1}^{N_t'} y_k' \tag{22}$$

The parameters of Ψ are optimized by minimizing the objective function as follows:

$$Q_2' = \sqrt{\frac{1}{N_t'}\sum_{k=1}^{N_t'}\left(\sum_{i=1}^{c} A_i(\boldsymbol{x}',\boldsymbol{v}_i)\Psi_i(L_i(\boldsymbol{x}',\boldsymbol{a}_i)) - y_k'\right)^2} + \frac{\lambda}{2} w^T w \tag{23}$$

The construction of mapping Ψ is similar to the construction of mapping Φ in Method 1. $\Psi = [\Psi_1\ \Psi_2\ \cdots\ \Psi_c]$, and the parameters are obtained by minimizing (23) using the labeled dataset $\boldsymbol{H}_L$; w represents the vector of all the parameters optimized.

Step 2c) Method 3: changing both the input and output spaces

In this case, both the conditions and the conclusions of the fuzzy rules in the two domains are different. The target domain's ideal model is described in (9).

This method is a combination of the first two and uses the mappings to modify the input and output spaces. The input space is transformed to $\Phi(\boldsymbol{x}')$ by mapping Φ, the output space is transformed by mapping Ψ, and the new fuzzy model M_3' for the target domain is constructed based on the fuzzy rules in model M. The process and resulting architecture is shown in Fig. 7.

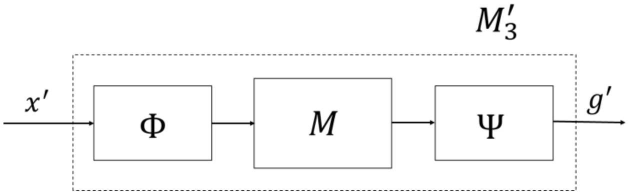

Fig. 7. Method 3: changing both the input and output spaces

Model M_3', described in the form of fuzzy rules, is:

$$\text{if } \boldsymbol{x}' \text{ is } A_i(\Phi(\boldsymbol{x}'),\Phi(\boldsymbol{v}_i)), \text{ then } g' \text{ is } \Psi_i(L_i(\boldsymbol{x}',\boldsymbol{a}_i) \\ i = 1,\dots,c \tag{24}$$

Therefore, the output of model M_3' is:

$$g' = \sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'),\Phi(\boldsymbol{v}_i))\ \Psi_i(L_i(\Phi(\boldsymbol{x}'),\boldsymbol{a}_i)) \tag{25}$$

Our aim is to find a Φ and Ψ such that $M_3' \approx \widetilde{M}_3$, i.e.,

$$\sum_{k=1}^{N_t'} g_k' \approx \sum_{k=1}^{N_t'} y_k' \tag{26}$$

i.e.

$$\sum_{k=1}^{N_t'}\sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'),\Phi(\boldsymbol{v}_i))\ \Psi_i(L_i(\Phi(\boldsymbol{x}'),\boldsymbol{a}_i)) \approx \sum_{k=1}^{N_t'} y_k' \tag{27}$$

The parameters of Φ and Ψ are optimized by minimizing the objective function as follows:

$$Q_3' = \sqrt{\frac{1}{N_t'}\sum_{k=1}^{N_t'}\left(\sum_{i=1}^{c} A_i(\Phi(\boldsymbol{x}'),\Phi(\boldsymbol{v}_i)\Psi(L_i(\Phi(\boldsymbol{x}'),\boldsymbol{a}_i)) - y_k'\right)^2} + \frac{\lambda}{2} w^T w \tag{28}$$

The construction of mappings Φ and Ψ is exactly the same as that in Methods 1 and 2, and the parameters of Φ and Ψ are optimized by minimizing (28) using the labeled dataset $\boldsymbol{H}_L$. Similarly, the objective function includes two terms: the approximation error and the structural risk.

B. Performance Index

Another model is also trained using insufficient data in the target domain. Although there is only a small amount of labeled data in the target domain, they can still be used to train a model. Proving that a model does not perform as well when trained with less data in the target domain supports our assumption. As a result, three models are constructed: the first is built using the source data for the source domain (model M); the second is built using the insufficient target data for the target domain (model $\overline{M}$); and the third is built using the proposed granular fuzzy domain adaptation methods (models M_1', M_2', M_3', corresponding to three cases).

The datasets in the source domain and the target domain are $\boldsymbol{S}$ and $\boldsymbol{H}$, as described in the above subsection. When constructing the above models, we used a five-fold cross validation procedure, which is commonly used to validate models in machine learning. Dataset $\boldsymbol{S}$ is split into a training set $\boldsymbol{S}_1$(80%) and a testing set $\boldsymbol{S}_2$(20%). The number of data in $\boldsymbol{S}$, $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ are N, N_1, and N_2, respectively. Similarly, the labeled data in $\boldsymbol{H}_L$ are split into a training set $\boldsymbol{H}_{L1}$ (80%) with N'_{t1} data and a testing set $\boldsymbol{H}_{L2}$ (20%) with N'_{t2} data.

Symbolic representations of the models' performance follow.

Model's performance in the source domain is represented by Q, which is the root mean square error of the expected value and the output from model M.

$$Q = \sqrt{\frac{1}{N_2}\sum_{k=1}^{N_2}(d_k - y_k)^2} \tag{29}$$

where d_k is the output of model M when its input is $\boldsymbol{x}_k$, $\boldsymbol{x}_k \in \boldsymbol{S}_2$.

For consistency, dataset $\boldsymbol{H}_U$ is used to test the model's performance in the target domain in the following discussion.

The performance of model M in the target domain is denoted by Q_1, which indicates the ability of the source domain's model to address tasks in the target domain.

$$Q_1 = \sqrt{\frac{1}{N'-N'_t}\sum_{k=N'_t+1}^{N'}(h'_k - y'_k)^2} \tag{30}$$

where h'_k is the output of model M when the input is $\boldsymbol{x}'_k$, $\boldsymbol{x}'_k \in \boldsymbol{H}_U$, and y'_k is the expected output to input $\boldsymbol{x}'_k$.

The insufficient labeled data in the target domain $\boldsymbol{H}_L$ are used to train a model $\bar{M}$ for the target domain using the same construction procedures as model M for the source domain. The performance of model $\bar{M}$ in the target domain is denoted as Q_2:

$$Q_2 = \sqrt{\frac{1}{N'-N'_t}\sum_{k=N'_t+1}^{N'}(s'_k - y'_k)^2} \tag{31}$$

where s'_k is the output of model $\bar{M}$ when its input is $\boldsymbol{x}'_k$, $\boldsymbol{x}'_k \in \boldsymbol{H}_U$, and y'_k is the expected output to input $\boldsymbol{x}'_k$.

The model $M'(M'_1, M'_2, M'_3)$, constructed using our GFRDA methods, is also tested on the target dataset $\boldsymbol{H}_U$, and the result is denoted as Q_3:

$$Q_3 = \sqrt{\frac{1}{N'-N'_t}\sum_{k=N'_t+1}^{N'}(t'_k - y'_k)^2} \tag{32}$$

where t'_k is the output of model M' when the input is $\boldsymbol{x}'_k$, $\boldsymbol{x}'_k, \in \boldsymbol{H}_U$, y'_k is the expected output to input $\boldsymbol{x}'_k$.

When constructing the model M' for the target domain, a differential evolution (DE) optimization algorithm is used to optimize the parameters of the mappings and build the new GFRDA models for the target domain. DE is a computational method that determines an optimal solution by iteratively navigating a population of solutions, which minimizes a certain predetermined objective function. Such methods are commonly known as metaheuristics, as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions [31, 32]. Particle swarm optimization (PSO) is another famous evolutionary algorithm. Based on the experimental results from our previous paper [27], the algorithmic stability of DE is superior to PSO, so DE was selected as the optimization algorithm for the models' construction. In DE, there are two parameters that largely influence optimization performance: the differential weight F and the crossover probability CR. The value range of F is $[0,2]$, and the value range of CR is $[0,1]$. In addition, due to the problem's complexity, the same initialization strategy is used in all the experiments below: 200 candidate solutions are generated, and the maximum number of iterations is set to 200.

The values of Q_1, Q_2 and Q_3 are compared in the following experiments. The desired outcome is that Q_3 should be smaller than both Q_1 and Q_2 . $Q_3 < Q_1$ demonstrates that the performance of the new constructed model M' on the target domain is superior to the existing model M, and $Q_3 < Q_2$ shows that the model $\bar{M}$ trained using a few labeled data has poor performance compared to model M'.

V. Experiments on Synthetic Data

Both synthetic and real-world datasets were used to evaluate the proposed GFRDA methods and their algorithms. These datasets are described in this and the following section.

This section consists of three experiments to discuss and analyze the effectiveness of the proposed GFRDA methods. The first experiment validates the presented methods and analyzes the impact of an important parameter in the performance of the constructed models – the trade-off parameter λ. The second experiment explores the effect on the results when the number of the labeled target data changes. The third experiment compares the outcomes of the three methods when dealing with different cases in domain adaptation problems.

The datasets used in these three experiments contain some repetition, so all the datasets are described first, followed by the details of their application.

The datasets are generated by the input data and the linear functions. As described in Section II, the conditions of the fuzzy rules are governed by the centers of the clusters, which decide the membership functions of the constructed fuzzy sets. Since FCM is used to build the clusters and the fuzzy sets, the cluster centers are significantly affected by the distribution of the input data. Therefore, to obtain the source and target data with different cluster centers, the input data in the source and target domains should be generated with different distributions.

Two groups of input data with different distributions are shown in Table I, and similarly two groups of linear functions with disparate parameters are displayed in Table II.

Table I contains two groups of input data, input data 1 and input data 2; the method used to generate the data was the same for both groups. To obtain more than one fuzzy rule and differentiate between the cluster centers, we generated three sub-datasets by varying the distribution and combining them to construct the whole dataset. The first sub-dataset was generated using normalized distribution $N(\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1)$, and the other two sub-datasets were built in the same way.

Table II, lists the two groups of linear functions with different parameters. There are three linear functions in each group that correspond to the input data in Table I.

TABLE I
THE DISTRIBUTIONS OF INPUT DATA 1 AND DATA 2

Input data 1		Input data 2	
Mean values	Covariance matrixes	Mean values	Covariance matrixes
$\boldsymbol{\mu}_1 = [1\ 1]$	$\boldsymbol{\sigma}_1 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	$\boldsymbol{\mu}'_1 = [2.5\ 1.5]$	$\boldsymbol{\sigma}'_1 = \begin{bmatrix} 0.4^2 & 0.1 \\ 0.1 & 0.4^2 \end{bmatrix}$
$\boldsymbol{\mu}_2 = [2\ 1]$	$\boldsymbol{\sigma}_2 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	$\boldsymbol{\mu}'_2 = [2\ 1.5]$	$\boldsymbol{\sigma}'_2 = \begin{bmatrix} 0.5^2 & 0.1 \\ 0.1 & 0.5^2 \end{bmatrix}$
$\boldsymbol{\mu}_2 = [1.5\ 2]$	$\boldsymbol{\sigma}_3 = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}$	$\boldsymbol{\mu}'_3 = [2.5\ 2]$	$\boldsymbol{\sigma}'_3 = \begin{bmatrix} 0.5^2 & 0.1 \\ 0.1 & 0.5^2 \end{bmatrix}$

TABLE II
COEFFICIENTS OF LINEAR FUNCTIONS IN THE TWO DIFFERENT GROUPS

Linear functions 1		Linear functions 2	
$L_1(a_1)$	$a_1 = [1\quad 1\quad 1]$	$L_1(a'_1)$	$a'_1 = [2\quad 0.5\quad 1.5]$
$L_2(a_2)$	$a_2 = [2\quad 2\quad 1]$	$L_2(a'_2)$	$a'_2 = [1\quad 2\quad 0.5]$
$L_3(a_3)$	$a_3 = [-1\quad 1\quad 3]$	$L_3(a'_3)$	$a'_3 = [-1.5\quad 2\quad 4.5]$

Various combinations of the input data in Table I and the linear functions in Table II result in different datasets. Thus, three datasets were constructed, as shown in Table III, to represent the three different cases in domain adaptation described in Section III.

TABLE III
THREE DATASETS THAT REPRESENT THREE DIFFERENT CASES IN DOMAIN ADAPTATION

	Source domain	Target domain
Datasets 1	Input data 1+ Linear functions 1	Input data 2 + Linear functions 1
Datasets 2	Input data 1+ Linear functions 1	Input data 1 + Linear functions 2
Datasets 3	Input data 1 + Linear functions 1	Input data 2 + Linear functions 2

From Table III, we can see that the dataset in the source domain is fixed, and the varying dataset in the target domains lead to three different cases. Constructing the target data using input data 2 and linear function 1 reflects cases where the input data differs between the two domains. Using input data 1 and linear function 2, reflects cases where the conclusion of the fuzzy rules are not the same, and using input data 2 and linear function 2 reflects differences in both the conditions and conclusions.

A. Verifying the Proposed GFRDA Methods

The purpose of this subsection is to verify the ability of the proposed methods to solve three cases in domain adaptation, and further explore the impact of the parameter λ on the performance of the models.

Three experiments were conducted to test the methods' performance in the different domain adaption cases using the three datasets in Table III. Additionally, comparing the models' performance with varying parameter λ was used to determine the optimal λ. There are 1500 labeled data in the source domain, and 15 labeled and 585 unlabeled data in the target domain.

The results and analysis of these three experiments are discussed in detail below.

a) Method 1: change the input space

This experiment changed the input space using Dataset 1 from Table III to deal with domain adaptation cases where the fuzzy rule conditions differ. Moreover, comparing model performance with different values for parameter λ was used to determine the optimal λ. The results are shown in Table IV.

TABLE IV
THE EXPERIMENTAL RESULTS OF THE FIRST METHOD BY VARYING λ

λ	Q	Q_1	Q_2	Q_3
0	0.08 ± 0.01	1.88 ± 0.01	77730.09 ± 163972.14	1.13 ± 0.22
0.1				1.06 ± 0.07
0.2				1.04 ± 0.06
0.3				1.04 ± 0.05
0.4				1.04 ± 0.04
0.5				1.05 ± 0.04
0.6				1.06 ± 0.03
1				1.15 ± 0.04
2				3.41 ± 0.38

Because five-fold cross validation was used, all the values for Q, Q_1, Q_2 and Q_3 are written in the form of "mean ± standard deviation". Since changing λ only impacts the construction of model M'_1, the values of Q, Q_1 and Q_2 that are related to model M and $\bar{M}$ are constant under different λ values. From Table IV we can see that the mean value of Q_1 is 1.88, which indicates that the model of the source domain does not fit the target data very well. The number of labeled target data is small, resulting in a very large mean value of 77730.09 and a large standard deviation of 163972.14 for Q_2 , to represent model M''s performance. However, when λ is not bigger than 1, the mean values for Q_3 are smaller than those of Q_1 and Q_2. This indicates that the model built using our method is superior to the source domain's model and the model constructed using the target data. When λ is greater than 0.4, the values of Q_3 appear to have a growth trend and is lowest when λ is equal to 0.4.

b) Method 2: change the output space

Dataset 2 was used to simulate cases where the conclusions of the fuzzy rules differ by changing the output space. The results are shown in Table V.

TABLE V
THE EXPERIMENTAL RESULTS OF THE SECOND METHOD BY VARYING λ

λ	Q	Q_1	Q_2	Q_3
0	0.50 ± 0.02	3.37± 0.00	62496.40± 122924.89	1.84 ± 0.28
0.01				1.85 ± 0.24
0.02				1.89 ± 0.26
0.05				2.06 ± 0.30
0.1				3.06 ± 0.29
1				8.90 ± 0.08

Compared to the last experiment, the values of Q_3 are sensitive to changes in λ, and tends to increase with an increase in λ. When λ is smaller than 0.1, the mean values of Q_3 are no greater than the mean values of Q_1 and Q_2, which shows the superiority of our proposed method. Model M_1' shows the best performance when λ is set to 0.

c) Method 3: change the input and output spaces

Dataset 3 was used to test cases where both the conditions and conclusions of the fuzzy rules differ; therefore, both the input and output spaces were transformed. The results are shown in Table VI.

TABLE VI
THE EXPERIMENTAL RESULTS OF THE THIRD METHOD BY VARYING λ

λ	Q	Q_1	Q_2	Q_3
0	0.08± 0.01	3.28± 0.02	62496.40± 122924.89	2.32± 0.37
0.01				2.85± 0.10
0.02				2.92± 0.29
0.05				3.448± 0.30
0.1				4.05± 0.19
1				9.00± 0.07

From Table VI, we can see that similar to the last experiment, a tiny change in λ results in and increase in Q_3. When λ is not smaller than 0.05, the mean value of Q_3 is greater than that of Q_1, which means the model using the proposed method is not better than the source domain model. However, the proposed method works well when the value of λ is small.

B. Exploring the Impact of the Number of Labeled Target Data

In the above experiments, the number of labeled target data was fixed at 15. Since the optimization of the models $M_1', M_2',$ and M_3' is totally based on labeled target data, they play an important role in the ability of the constructed model to fulfill the target tasks.

This experiment was designed to analyze the performance of the constructed model with different numbers of labeled target data. The total number of data in the target domain is 1500, but the number of labeled data varies.

Here, we only transform the input space, as an example and list the results in Table VII.

TABLE VII
THE VALUES OF Q, Q_1, Q_2 AND Q_3 WITH DIFFERENT NUMBERS OF LABELED TARGET DATA

N_t'	Q	Q_1	Q_2	Q_3
10	0.08 ± 0.01	1.88 ± 0.01	507710.47 ± 940131.29	1.07 ± 0.07
15		1.88 ± 0.01	77730.09 ± 163972.14	1.04 ± 0.05
20		1.89 ± 0.01	17.23 ± 33.47	1.04 ± 0.04
25		1.88 ± 0.01	0.85 ± 0.54	1.01 ± 0.03
30		1.89 ± 0.01	0.76 ± 0.12	1.02 ± 0.03

As Q represents the performance of the source model on the source data, changing the number of labeled target data N_t' has no impact; the value of Q is constant at different N_t'. The values of Q_1 . which represents the performance of source model on unlabeled target data with number $N' - N_t'$, have tiny fluctuations, which indicates that changes in N_t' only slightly influence Q_1. The mean value and standard deviation of Q_2 decrease with a greater amount of labeled target data. This is because more training data is available in the target data, and model $\bar{M}$ is able to achieve better generalization of the unlabeled target data. Even though few labeled target data are available, the values for Q_3 are smaller than that of Q_1 and Q_2, which indicates that our proposed method works well in this domain adaptation problem. When the number of labeled target data is beyond 25, the proposed method does not show superiority. However, given our central assumption that the labeled target data are insufficient to construct a good model, the results obtained are reasonable.

C. Comparing the Performance of the Proposed Methods

Experiments 1 and 2 show that each of the proposed methods are effective solutions to their respective domain adaptation problems. However, we were also curious about each method's ability to solve the other two cases and designed an experiment to compare the performance of all three methods in all three cases.

We must highlight that the purpose of this experiment is not to determine the best method for each case. First, the performance of these methods depends heavily on the datasets, so results from one dataset do not prove the validity of these methods in the given case. Second, we have already proven that each case is well-handled by its specifically designed method.

However, if the either of the other two methods are also effective, they can be treated as 'assistant' methods.

The three methods are used to solve the three cases in domain adaptation problems, and the results are displayed in Table VIII.

TABLE VIII
COMPARISON OF THE THREE METHODS USED IN SELECTED ADAPTATION CASES

	Case 1	Case 2	Case 3
Q	0.08 ± 0.01	0.50 ± 0.02	0.08 ± 0.01
Q1	1.88 ± 0.01	3.37 ± 0.00	3.28 ± 0.02
Q2	77730.09 ± 163972.14	62496.40 ± 122924.89	62496.40 ± 122924.89
Q3(method 1)	1.04 ± 0.05	4.00 ± 0.17	3.07 ± 0.14
Q3(method 2)	1.01 ± 0.18	1.84 ± 0.28	1.85 ± 0.13
Q3(method 3)	1.22 ± 0.32	1.88 ± 0.16	2.32 ± 0.37

From the results shown in Table VIII, we can see that all three methods perform well in the first case. In the second case, changing the output space or changing both the input and output spaces can also solve this domain adaptation problem. Similarly, all three methods are also valid for the third case, just not as well as the other two.

Based on the results, we conclude that the specific method designed for each case show superior performance; furthermore, the other methods can be used as alternatives.

VI. EXPERIMENTS ON REAL-WORLD DATASETS

Three real-world datasets from the UCI Machine Learning Repository were used to validate the effectiveness of the proposed GFRDA methods. However, information about which case each datasets reflects is not readily available, so we use the three methods to solve this problem and discover which method was the most effective.

The "concrete compressive strength" dataset contains eight input features to predict the concrete compressive strength output feature. The dataset was revised in two ways to make it appropriate for use in a transfer learning problem. First, the dataset was split into a source domain and a target domain based on the input feature "age": instances with an age smaller than 100 fell into the source domain, and the remaining instances were treated as data in the target domain. Second, the input features "blast furnace slag", "fly ash", and "superplasticizer" were perturbed with random numbers using the normal distributions $N(0.1, 0.1)$ in the source data and $N(5,1)$ for the target data. There are 900 labeled instances in the source domain, and 30 labeled and 80 unlabeled instances in the target domain.

The "housing dataset" aims to predict the "MEDV" 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" of the source data were perturbed by random numbers coming from $N(0.1, 0.1)$, while those attributes in the target data were perturbed by normal random numbers using the distributions $N(7,1)$, $N(5,1)$ and $N(8,1)$, respectively. There are 360 labeled instances in the source domain and 130 instances in the target data with 15 labeled.

The "Istanbul stock exchange" dataset aims to predict the "MSCI emerging markets index" using the attributes "stock exchange returns" and "Istanbul stock exchange national 100 index". The data was normalized and split into two datasets. The first 300 instances were used to form the source domain, and the next 120 instances were chosen as the target domain. Further, the two attributes were perturbed with random numbers using the normal distributions $N(0.1, 0.1)$ in the source data and $N(5,1)$ for the target data.

The last dataset concerns "air quality". From the provided attributes, we selected two attributes, "temperature" and "relative humidity", as the input data, and chose "absolute humidity" as the output. All the attributes were normalized, and the dataset was split into two domains based on "relative humidity". The data with a "relative humidity" of greater than 0.5 were chosen as the source domain, and the remaining data were used to form the target domain. Further, the two attributes in the source data were all perturbed by random numbers following a normal distribution $N(0.1, 0.1)$, and the two attributes in the target data were perturbed by the normal random numbers following $N(7,1)$ and $N(5,1)$ respectively. There are 3600 labeled instances in the source domain and 1200 instances in the target data with 15 labeled.

Five-fold cross validation was used for all experiments, and the results are shown in Table IX.

TABLE IX
EXPERIMENTAL RESULTS FROM THE REAL-WORLD DATASETS

	Concrete compressive strength	Housing	Istanbul stock exchange	Air quality
Q	0.11 ± 0.02	0.11 ± 0.01	0.09 ± 0.04	0.13 ± 0.02
Q1	1.07 ± 0.16	2.35 ± 0.51	3.05 ± 0.70	6.41 ± 0.35
Q2	6488.52 ± 4938.04	5.94 ± 6.29	76627.04 ± 170157.35	7.27 ± 15.75
Q3 (method 1)	0.18 ± 0.06	0.60 ± 0.79	0.12 ± 0.00	0.15 ± 0.00
Q3 (method 2)	0.15 ± 0.01	0.18 ± 0.06	0.13 ± 0.00	0.15 ± 0.01
Q3 (method 3)	0.91 ± 1.57	0.19 ± 0.10	0.15 ± 0.01	0.15 ± 0.00

From the results, we can see that the mean values of Q_3 in all three methods are all smaller than the mean values for Q_1 and Q_2. This indicates that the models constructed using the proposed methods are better than both the existing source domain model and the model built using few labeled target data. The first and second methods build well-performing models for the target domain using the "concrete compressive strength" dataset. Compared to the first method, the second and third methods did a good job transferring the knowledge from the source domain to the target domain in the “housing” dataset. All the three methods work well in the “Istanbul stock exchange” dataset, and the first method showed a slight lead. On the “air quality” dataset, the three methods showed similar results and all worked well in addressing this domain adaptation problem.

VII. Conclusions and Future Work

In this study, we propose three granular fuzzy regression domain adaptation methods to address three challenging cases in fuzzy domain adaptation: where the conditions, conclusions, or both the conditions and conclusions of the fuzzy rules in the source and targets domains differ. These methods modify the input and/or output of the data space through mappings to make the fuzzy rules of the existing model more compatible for solving tasks in the target domain. Our methods effectively solve regression problems in the target domain even when only a small amount of labeled data is available. Experimental results show that the proposed methods greatly improve over the performance of existing models in estimating the values of the target data.

Our future studies will focus on cross-domain adaptation problems. These are more complicated than domain adaptation problems, where the input spaces of two domains have the same dimensionality. Cross-domain adaptation studies knowledge transfer in two domains that have different feature spaces, and a dissimilar number of fuzzy rules. This paper is an initial step in using the knowledge of GrC to improve the performance of prediction models for the target domain. More ideas and methods that use GrC will be applied to facilitate knowledge transfer in our further studies.

Acknowledgment

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

References

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

[2] H. Al-Mubaid and S. A. Umair, "A new text categorization technique using distributional clustering and learning logic," *IEEE Transactions on Knowledge and Data Engineering,* vol. 18, pp. 1156-1165, 2006.

[3] K. Sarinnapakorn and M. Kubat, "Combining subclassifiers in text categorization: A dst-based solution and a case study," *IEEE Transactions on Knowledge and Data Engineering,* vol. 19, pp. 1638-1651, 2007.

[4] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4460-4464.

[5] S. Sun, H. Shi, and Y. Wu, "A survey of multi-source domain adaptation," *Information Fusion,* vol. 24, pp. 84-92, 2015.

[6] G. Zhou, Y. Zhou, X. Guo, X. Tu, and T. He, "Cross-domain sentiment classification via topical correspondence transfer," *Neurocomputing,* vol. 159, pp. 298-305, 2015.

[7] N. Azizi, N. Farah, and M. T. Khadir, "Multi-stage dynamic ensemble selection using heterogeneous learning algorithms: application on classification problems," *International Journal of Knowledge Management Studies,* vol. 6, pp. 16-30, 2015.

[8] 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.

[9] D. C. Cireşan, U. Meier, and J. Schmidhuber, "Transfer learning for Latin and Chinese characters with deep neural networks," in *2012 International Joint Conference on Neural Networks (IJCNN)*, 2012, pp. 1-6.

[10] R. Luis, L. E. Sucar, and E. F. Morales, "Inductive transfer for learning Bayesian networks," *Machine Learning,* vol. 79, pp. 227-255, 2010.

[11] V. Behbood, J. Lu, and G. Zhang, "Text categorization by fuzzy domain adaptation," in *2013 IEEE International Conference on Fuzzy Systems (FUZZ)*, 2013, pp. 1-7.

[12] P. Swietojanski, A. Ghoshal, and S. Renals, "Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR," in *Spoken Language Technology Workshop (SLT), 2012 IEEE*, 2012, pp. 246-251.

[13] A. Ahmed, K. Yu, W. Xu, Y. Gong, and E. Xing, "Training hierarchical feed-forward visual recognition models using transfer learning from pseudo-tasks," in *European Conference on Computer Vision*, 2008, pp. 69-82.

[14] K. Ueki, M. Sugiyama, and Y. Ihara, "Perceived age estimation under lighting condition change by covariate shift adaptation," in *2010 20th International Conference on Pattern Recognition (ICPR)*, 2010, pp. 3400-3403.

[15] D. L. Silver and R. E. Mercer, "Sequential inductive transfer for coronary artery disease diagnosis," in *International Joint Conference on Neural Networks*, 2007, pp. 2635-2641.

[16] 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.

[17] V. Behbood, J. Lu, and G. Zhang, "Text categorization by fuzzy domain adaptation," in *IEEE International Conference on Fuzzy Systems (FUZZ),*, 2013, pp. 1-7.

[18] 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.

[19] 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.

[20] 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, pp. 1917-1935, 2015.

[21] J. Shell and S. Coupland, "Towards fuzzy transfer learning for intelligent environments," in *International Joint Conference on Ambient Intelligence*, 2012, pp. 145-160.

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

[23] 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.

[24] Z. Deng, Y. Jiang, H. Ishibuchi, K.-S. Choi, and S. Wang, "Enhanced Knowledge-Leverage-Based TSK Fuzzy System Modeling for Inductive Transfer Learning," *ACM Transactions on Intelligent Systems and Technology (TIST),* vol. 8, p. 11, 2016.

[25] Y.-J. Liu, Y. Gao, S. Tong, and Y. Li, "Fuzzy approximation-based adaptive backstepping optimal control for a class of nonlinear discrete-time systems with dead-zone," *IEEE Transactions on Fuzzy Systems,* vol. 24, pp. 16-28, 2016.

[26] J. Qiu, S. X. Ding, H. Gao, and S. Yin, "Fuzzy-model-based reliable static output feedback control of nonlinear hyperbolic PDE systems," *IEEE Transactions on Fuzzy Systems,* vol. 24, pp. 388-400, 2016.

[27] 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,* 2016.

[28] M. L. Hadjili and V. Wertz, "Takagi-Sugeno fuzzy modeling incorporating input variables selection," *IEEE Transactions on Fuzzy Systems,* vol. 10, pp. 728-742, 2002.

[29] W. Pedrycz, "Allocation of information granularity in optimization and decision-making models: towards building the foundations of granular computing," *European Journal of Operational Research,* vol. 232, pp. 137-145, 2014.

[30] W. Pedrycz, *Granular computing: Analysis and design of intelligent systems*: CRC Press, 2013.

[31] K. Price, R. M. Storn, and J. A. Lampinen, *Differential evolution: a practical approach to global optimization*: Springer Science & Business Media, 2006.

[32] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization,* vol. 11, pp. 341-359, 1997.

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, Centre for Artificial Intelligence, University of Technology Sydney.

Guangquan Zhang is an Associate Professor and Director of the Decision Systems and e-Service Intelligent (DeSI) Research Laboratory in Center for AI, 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 seven 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 Distinguished Professor, Director of Centre for Artificial Intelligence and the Associate Dean in Research Excellence with the Faculty of Engineering and Information Technology at the University of Technology Sydney, Australia. She received her Ph.D. degree in information systems from Curtin University of Technology, Australia, in 2000.

Her main research expertise is in 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; awarded eight 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, general/PC/organization chairs for ten international conferences as well as having delivered 16 keynote/ plenary speeches at IEEE and other international conferences.