Transfer Learning Under Conditional Shift Based on Fuzzy Residual

Gengxiang Chen, Yingguang Li, *Member, IEEE*, and Xu Liu

***Abstract*—Transfer learning has received much attention recently and has been proven to be effective in a wide range of applications, whereas studies on regression problems are still scarce. In this article, we focus on the transfer learning problem for regression under the situations of conditional shift where the source and target domains share the same marginal distribution while having different conditional probability distributions. We propose a new framework called transfer learning based on fuzzy residual (ResTL) which learns the target model by preserving the distribution properties of the source data in a model-agnostic way. First, we formulate the target model by adding fuzzy residual to a model-agnostic source model and reuse the antecedent parameters of the source fuzzy system. Then two methods for bias computation are provided for different considerations, which refer to two ResTL methods called ResTL_{LS} and ResTL_{RD}. Finally, we conduct a series of experiments both on a toy example and several real-world datasets to verify the effectiveness of the proposed method.**

***Index Terms*—Conditional shift, fuzzy residual, fuzzy system, transfer learning.**

I. Introduction

SYSTEMS based on traditional machine learning techniques often face a major challenge when applied in real-world applications. It is expensive or even impossible to collect a large volume of labeled training data [1], [2]. To address this challenge of data scarcity, transfer learning, which can enhance the learning ability in a target domain by transferring the information from related domains, has received much attention recently and has been proven to be effective in a wide range of applications [3]. Under this powerful paradigm, various learning methods were proposed according to different assumptions about the relationship between source and target domains, including covariate shift [4], [5]; prior probability shift [6], [7]; sample selection bias [8], [9]; conditional shift [10]; and so on. In this article, we are mainly concerned with the transfer learning problem for regression under the situations of conditional shift, which is a common and challenging data shift problem. A conditional shift in the multistream learning problem has received great attention for a long time [11], [12]. As the nature of data streams, both source and target streams are subject to their own concept drift over time: $P_S(y|\mathbf{x})_t \neq P_S(y|\mathbf{x})_{t-1}$, $P_T(y|\mathbf{x})_t \neq P_T(y|\mathbf{x})_{t-1}$ [13]. However, in this article, we focus on the conditional shift across domains rather than intradomain, where the source and target domains have different conditional distributions, while sharing the same marginal distribution, that is, $P(y_s|\mathbf{x}_s) \neq P(y_t|\mathbf{x}_t)$, whereas $P(\mathbf{x}_s) = P(\mathbf{x}_t)$.

In the engineering field, many kinds of training data need to be collected and labeled by ineffective manual measurement with expensive instruments [14], [15]. Fig. 1 shows an example to construct the prediction model of pose-dependent tooltip dynamics of a computer numerical control (CNC) machine center. Tool tip dynamics, including natural frequency w, damping ratio ξ, and stiffness K, are very important for chatter suppression in machining complex parts like an aero-engine impeller. In the machine's workspace defined by three linear axes (X, Y, Z) and two rotating axes (A, C), (w, ξ, K) usually varies at different positions thus a model f that maps (X, Y, Z, A, C) to (w, ξ, K) is required. Since the workspace is huge, for each tool-holder assemble the machine has to be turned off for a couple of weeks to carry out a large amount of impact testing experiments manually for label collection [16]. Moreover, a CNC machine usually has more than one hundred of frequently used tool-holder assembles, it is impossible to collect enough labeled tooltip dynamics data in real manufacturing industry. Although the data distributions produced by different subsetting of a system are different, for example, the tooltip dynamics distributions of different tool-holder assembles discussed in Fig. 1, they are usually similar. By transferring the knowledge learned from previous subsetting during the training process of a target subsetting, the required labeled data can be greatly reduced. More specifically, the data generated under different subsetting of a system usually have the same marginal distributions but different conditional distributions, thus transfer learning among this kind of data can be categorized into conditional shift problem. There are many other regression problems under the conditional shift in real industrial applications, including kinematic parameters prediction of robot with different end effectors [17] and tool wear estimation [18]. And it is also a common issue in many other fields, such as the predictions of the yield of

Manuscript received October 9, 2019; revised February 6, 2020; accepted April 8, 2020. Date of publication May 20, 2020; date of current version February 16, 2022. This work supported by the National Natural Science Foundation Project of China under Grant 51925505, Grant U1537209, and Grant 51605217. This article was recommended by Associate Editor C.-F. Juang. *(Corresponding author: Yingguang Li.)*

Gengxiang Chen and Yingguang Li are with the National Key Laboratory of Science and Technology on Helicopter Transmission, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China (e-mail: cgx@nuaa.edu.cn; liyingguang@nuaa.edu.cn).

Xu Liu is with the School of Mechanical and Power Engineering, Nanjing Tech University, Nanjing 211816, China (e-mail: liuxu.smpe@njtech.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TCYB.2020.2988277.

Digital Object Identifier 10.1109/TCYB.2020.2988277

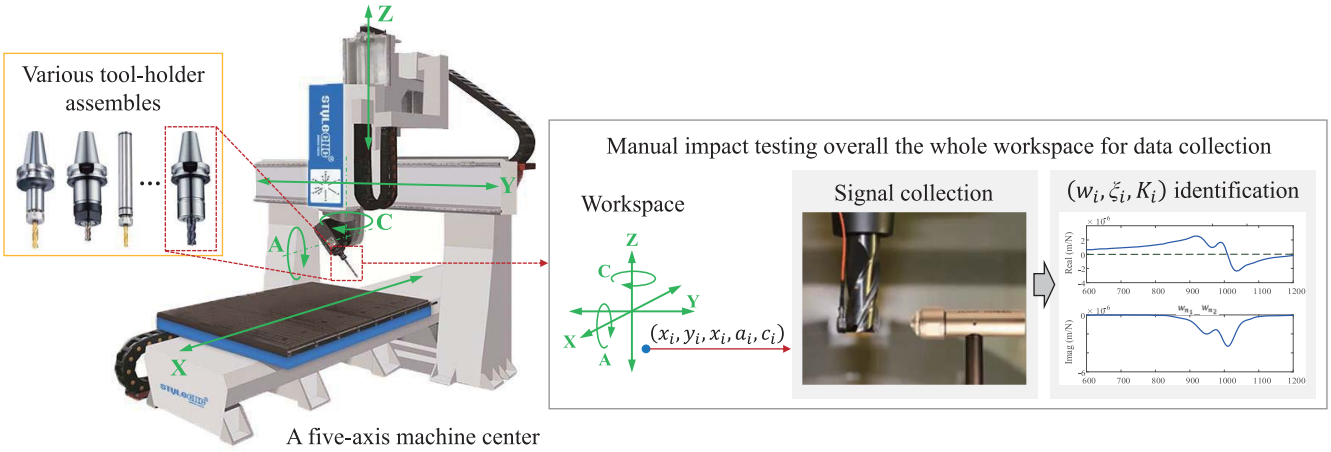


Fig. 1. Case of data labeling for tooltip dynamics prediction in CNC machining.

vineyards [19], the delay of particular flight [20], and the particulate matter (PM2.5) data [21].

To bridge different domains under substantial distribution discrepancy, great efforts have been made recently. Feature matching methods like transfer component analysis (TCA) [22] and joint distribution adaptation (JDA) [23] were proposed to extract domain invariant information for the covariate shift problem. Other methods, including kernel mean matching (KMM) [8], Kullback–Leibler importance estimation procedure (KLIEP) [26], and domain adaptation under generalized target shift (GeTarS) [10], tried to resample the source instances by reweighting or transforming to deal with covariate shift or prior probability shift problems. Besides, deep neural networks were also explored in this area as its excellent feature extraction capability [24]. For the conditional shift problem discussed in this article, existing work can be summarized into two main categories which are instance-based methods and model-based methods.

Instance-based methods are widely used for distribution shift correction and can be outlined into two scenarios, namely, instance reweighting and instance transformation. In *instance reweighting* methods, each instance in the source domain will be weighted to indicate its contribution in constructing the target model. The weights can be predetermined by minimizing the distribution discrepancy between the source and target data [8], [20], [26], and can also be adjusted adaptively according to the training error of the two datasets, like TrAdaBoost [27]–[29]. To guarantee the success, this kind of methods usually requires the distributions across two domains are very close in some connected sets of $\mathbf{x} \in \mathcal{X}$, then the larger weights can be assigned to the source instances in these sets [20]. However, this assumption may not be always satisfied because many real-world cases have global conditional shifts, that is, $P(y_s|\mathbf{x}) \neq P(y_t|\mathbf{x})$ for all $\mathbf{x} \in \mathcal{X}$. *Instance transformation* methods try to transform the source data to enclose the conditional distribution to the target data, that is, $P_s^{\text{new}} = P_t$ [19], [30]. Nevertheless, if the target data are inadequate for estimating the distribution, it is difficult to solve the coefficients of the transformation accurately [26].

Two kinds of *model-based* transfer learning methods are also investigated for conditional shift situations, namely, parameter-based methods and hypothesis-based methods. *Parameter-based* methods are established based on the assumption that the models in the source and target domains have similar structures or share some common parameters. To make the idea work, the prediction model has to be specified beforehand. The Gaussian process (GP) regression model is a popular choice for parameter-based methods by transferring through shared parameters [31] and shared representation of instances [32], or learning a transfer covariance function [33]. Recently, a more intuitive and interpretable modeling method, Takagi–Sugeno–Kang (TSK) fuzzy system, attracted more attention. Target models can be learned by transferring the consequent parameters [35], [36], or mapping the label space under fuzzy rules [37]–[39]. Though parameter-based methods show promising advantages for some applications, the limitation of model-specific parameters makes them not applicable to many scenarios [40]. *Hypothesis-based* methods aim to incorporate the hypothesis learned in the source domain instead of estimating the target model directly [42]. Different frameworks, including the offset and scale, are proposed to transform the source model to the target model [19], [42]. However, since the transformation function is trained only on the target data, the final performance is highly dependent of the amount of the labeled target data. A major challenge in conditional shift problem is how to train a target model reasonably with very limited target data.

In this article, we propose a new framework called transfer learning based on fuzzy residual (ResTL) which learns the target model for regression problems under conditional shift situation by preserving the distribution properties of the source data in a model-agnostic way. In ResTL, the target hypothesis $h_t(\mathbf{x})$ is obtained by adding a residual function $r(\mathbf{x})$ to the source hypothesis $h_s(\mathbf{x})$. Traditional hypothesis-based methods train $r(\mathbf{x})$ on the target data directly. However, it is usually difficult to find a reasonable solution in a considerable hypothesis space if the target data are quite limited. In this article, we follow the idea that the target model trained by transfer learning would preserve the distribution properties of

the source data, especially when the target data are far from enough. Therefore, ResTL first models the source data into a TSK fuzzy system. Then, the fuzzy residual $r(\mathbf{x})$ is obtained by multiplying the antecedent parameters of the source fuzzy system and the biases trained on the target data. As the fuzzy partition could be regarded as reflections to the marginal distribution of $\mathcal{X}$, ResTL can learn a much more reasonable target model by preserving the data properties of the source task, even the target data are quite limited. Furthermore, we relax $h_s(\mathbf{x})$ from the TSK fuzzy system to a model-agnostic form, which means that the target model could be generalized as a combination of $h_s(\mathbf{x})$ trained by any supervised regression model and a fuzzy residual component $r(\mathbf{x})$.

In Section II, we will introduce the main idea of ResTL and the model-agnostic formulation of the target model. Then two algorithms for bias computation are provided to construct fuzzy residual for different considerations. In Section III, we will describe a series of experiments carried out on toy examples and several real-world datasets to verify the effectiveness of the proposed method. Finally, we will conclude this article in Section IV.

II. Model-Agnostic Transfer Learning Based on Fuzzy Residual

Suppose that we have a regression task with a large amount of training data $\mathcal{D}_s = \{(\mathbf{x}_1^s, y_1^s), \ldots, (\mathbf{x}_{n_s}^s, y_{n_s}^s)\}$ as a source task, where $\mathbf{x}_i^s \in \mathcal{X}_S$ is the data instance and $y_i^s \in \mathcal{Y}_S$ is the corresponding label, a continuous variable. There is also another regression task with a small amount of training data $\mathcal{D}_t = \{(\mathbf{x}_1^t, y_1^t), \ldots, (\mathbf{x}_{n_t}^t, y_{n_t}^t)\}$ as a target task, where the input $\mathbf{x}_i^t \in \mathcal{X}_T$ and $y_i^t \in \mathcal{Y}_T$ is the corresponding output. This article focuses on the transfer learning problem between the two tasks under conditional shift situation, where the marginal distributions are the same, $P(\mathbf{x}_s) = P(\mathbf{x}_t)$, but the conditional distributions are different, $P(y_s|\mathbf{x}_s) \neq P(y_t|\mathbf{x}_t)$.

There are two crucial procedures for ResTL, preserving the shared knowledge across domains and reducing the discrepancy across domains. First, the TSK fuzzy system of the source task h_{FS_s} is constructed using the source data. Given that there are only small amounts of labeled target data, the TSK fuzzy system of the target task h_{FS_t} cannot be constructed directly. We propose to reuse the antecedent information of h_{FS_s} and describe the discrepancy between two models by adding bias to the fuzzy rules. The details about the proposed ResTL are introduced in the rest of this section. First, we formulate the target model by adding fuzzy residual to a model-agnostic source model. Then two methods for bias computation are provided which refer to two ResTL methods called $\mathrm{ResTL}_{\mathrm{LS}}$ and $\mathrm{ResTL}_{\mathrm{RD}}$.

A. Formulating the Target Model With Bias of Fuzzy Rules

The target hypothesis $h_t(\mathbf{x})$ is obtained by adding a residual function $r(\mathbf{x})$ to the source hypothesis $h_s(\mathbf{x})$. First, we model the source data using the TSK fuzzy system. Then, the target model can be expressed with a model-agnostic source model and a fuzzy residual term.

1) Constructing the TSK Fuzzy System of the Source Task: For the source data $\mathcal{D}_s$, denote $\mathbf{X}^s \in \mathbb{R}^{n_s \times d}$ which is a matrix having the vectors of inputs and $\mathbf{y}^s \in \mathbb{R}^{n_s \times 1}$ which is a vector containing the output

$$\mathbf{X}^s = \left[\mathbf{x}_1^s, \mathbf{x}_2^s, \ldots, \mathbf{x}_{n_s}^s\right]^\top, \mathbf{y}^s = \left[y_1^s, y_2^s, \ldots, y_{n_s}^s\right]^\top. \quad (1)$$

The TSK fuzzy system can represent the nonlinear dynamical systems with TSK fuzzy rules with high degree of precision. Denote the TSK fuzzy system of the source task as FS_s, then the output of FS_s can be represented by a combination of series submodels as

$$h_{\mathrm{FS}_s}(\mathbf{x}) = \sum_{k=1}^{K} \lambda_k(\mathbf{x}) f^k(\mathbf{x}) \quad (2)$$

where K is the number of fuzzy rules, and $\lambda_k(\mathbf{x})$ is the normalized membership degree which can be calculated with antecedent parameters as Appendix A [43], [44]. The antecedent parameters are obtained by fuzzy partition, which can be regarded as reflections to the marginal distribution of $\mathcal{X}$. Because of the assumption that $P(\mathbf{x}_s) = P(\mathbf{x}_t)$, the antecedent parameters can be shared across domains to preserving the distribution information of the source domain. $f^k(\mathbf{x})$ is the kth fuzzy rule, namely, a linear function of the input variables.

Denote $\mathbf{p}^k \in \mathbb{R}^{(d+1)}$ as the linear coefficient vector of $f^k(\mathbf{x})$, then $\mathbf{p} = [\mathbf{p}^1\ \mathbf{p}^2 \cdots\ \mathrm{p}^K]^\top \in \mathbb{R}^{(d+1)K \times 1}$ is the consequent parameters of the TSK fuzzy system. The consequent parameters of FS_s can be solved with labeled dataset $\mathcal{D}_s$.

To solve the consequent $\mathbf{p}$, we can construct $\mathbf{X}_p^s \in \mathbb{R}^{n_s \times (d+1)K}$ as

$$\mathbf{X}_p^s = \left[\Gamma_1 \mathbf{X}_e^s, \Gamma_2 \mathbf{X}_e^s, \ldots, \Gamma_K \mathbf{X}_e^s\right] \quad (3)$$

where $\mathbf{X}_e^s = [\mathbf{X}^s,\ \mathbf{1}] \in \mathbb{R}^{n_s \times (d+1)}$ is the extended matrix by appending a unitary column to $\mathbf{X}^s$. $\Gamma_k \in \mathbb{R}^{n_s \times n_s}$ is a diagonal matrix having the normalized membership degree $\lambda_k(\mathbf{x}_i)$ as its ith diagonal element

$$\Gamma_k = \begin{bmatrix} \lambda_k(\mathbf{x}_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_k(\mathbf{x}_{n_s}). \end{bmatrix}. \quad (4)$$

Then, the consequent parameters $\mathbf{p} \in \mathbb{R}^{(d+1)K}$ can be solved with the least-square (LS) method as

$$\mathbf{p} = \left[\left(\mathbf{X}_p^s\right)^\top \mathbf{X}_p^s\right]^{-1} \left(\mathbf{X}_p^s\right)^\top \mathbf{y}^s. \quad (5)$$

Up till now, the TSK fuzzy system of the source task as h_{FS_s} is completely constructed and can be used to construct the target model.

2) Constructing the Target Model With Fuzzy Residual: Given that there are only small amounts of labeled target data, the TSK fuzzy system of the target task h_{FS_t} cannot be constructed directly. We propose to reuse the antecedent parameters of the source fuzzy system to preserve the distribution properties of the source data. For the labeled target data $\mathcal{D}_t$, denote $\mathbf{X}^t \in \mathbb{R}^{n_t \times d}$ and $\mathbf{y}^t \in \mathbb{R}^{n_s \times 1}$ as the input and output, respectively

$$\mathbf{X}^t = \left[\mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_{n_t}^t\right]^\top, \quad \mathbf{y}^t = \left[y_1^t, y_2^t, y_{n_t}^t\right]^\top. \quad (6)$$

For an instance $(\mathbf{x}_i^t, y_i^t)$ in $\mathcal{D}_t$, the output of FS_s is

$$h_{\mathrm{FS}_s}(\mathbf{x}_i^t) = \sum_{k=1}^{K} \lambda_k(\mathbf{x}_i^t) f^k(\mathbf{x}_i^t). \tag{7}$$

Since $P(y_s|\mathbf{x}) \neq P(y_t|\mathbf{x})$, it is obvious that $h_{\mathrm{FS}_s}(\mathbf{x}_i^t) \neq y_i^t$. Here, we define the target fuzzy system FS_t by appending a constant bias z_k to each fuzzy rule of FS_s as

$$h_{\mathrm{FS}_t}(\mathbf{x}) = \sum_{k=1}^{K} \lambda_k(\mathbf{x}) \left(f^k(\mathbf{x}) + z_k \right) \tag{8}$$

where the antecedent parameters $\lambda_k(\mathbf{x})(k = 1, \ldots, K)$ are reused to preserve the distribution properties of the source data while the bias $z_k(k = 1, \ldots, K)$ is defined to match the discrepancy of $P(y_s|\mathbf{x})$ and $P(y_t|\mathbf{x})$. $h_{\mathrm{FS}_t}(\mathbf{x})$ could be decomposed to the following representation as:

$$h_{\mathrm{FS}_t}(\mathbf{x}) = \sum_{k=1}^{K} \lambda_k(\mathbf{x}) f^k(\mathbf{x}) + \sum_{k=1}^{K} \lambda_k(\mathbf{x}) z_k. \tag{9}$$

The former component is the TSK fuzzy system of the source task, but can be easily replaced by any supervised regression model. Therefore, the target model obtained with ResTL can be generalized as

$$h_t(\mathbf{x}) = h_s(\mathbf{x}) + r(\mathbf{x}) \tag{10}$$

where $h_s(\mathbf{x})$ is the source model which is model agnostic and $r(\mathbf{x}) = \sum_{k=1}^{K} \lambda_k(\mathbf{x}) z_k$ is the fuzzy residual. After obtaining the antecedent parameters $\lambda_k(\mathbf{x})$ and $h_s(\mathbf{x})$, the task to learn the target model is to compute the bias z_k on the target data. After obtaining the antecedent parameters $\lambda_k(\mathbf{x})$ and $h_s(\mathbf{x})$, the task to learn the target model is transferred to compute the bias z_k. With the proper bias, the source data can be transformed with (10) to mimic the distribution of target data and facilitate learning on the target domain. Therefore, the bias parameters should be calculated with labeled target data.

B. Computing the Bias

We propose two methods to compute the bias z_k: 1) the LS method by empirical risk minimization and 2) residual defuzzification (RD). The bounds of fuzzy residual with the two methods are also analyzed.

1) Computing the Bias With the LS Method: From a probabilistic viewpoint, the estimation of conditional distribution $P(y|\mathbf{x})$ can be written as the hypothesis $h(\mathbf{x})$. The conditional distribution matching can be converted into minimizing the empirical risk of $h_t(\mathbf{x})$ on the target data. The bias z_k could be intuitively obtained through optimization algorithms by minimizing the risk of the hypothesis on $\mathcal{D}_t$. In general, the empirical risk is defined by the average loss function on the training set

$$R_{\mathrm{emp}}(h_t) = \frac{1}{n_t} \sum_{i=1}^{n_t} L\left(y_i^t, h_t(\mathbf{x}_i^t)\right). \tag{11}$$

The goal is to find a hypothesis $\hat{h}$ among a fixed class of functions $\mathcal{H}$ for which the risk $R_{\mathrm{emp}}(h_t)$ is minimal

$$\widehat{h_t} = \underset{h_t \in \mathcal{H}}{\operatorname{argmin}} R_{\mathrm{emp}}(h_t). \tag{12}$$

The hypothesis h_t is formulized as (9), therefore the bias vector $\mathbf{z}=[z_1, z_2, \ldots, z_K]^T$ can be represented as

$$\mathbf{z} = \underset{\mathbf{z} \in \mathbb{R}^{K \times 1}}{\operatorname{argmin}} \frac{1}{n_t} \sum_{i=1}^{n_t} L\left(y_i^t, h_t(\mathbf{x}_i^t)\right). \tag{13}$$

The loss function can be defined as the square error as

$$L\left(y_i^t, h_t(\mathbf{x}_i^t)\right) = \left[y_i^t - h_t(\mathbf{x}_i^t)\right]^2, \ i = 1, 2, \ldots, n_t. \tag{14}$$

Substituting (9) into (14), it follows that:

$$L\left(y_i^t, h_t(\mathbf{x}_i^t)\right) = \left[y_i^t - h_s(\mathbf{x}_i^t) - \sum_{k=1}^{K} \lambda_k(\mathbf{x}_i^t) z_k \right]^2. \tag{15}$$

Denote $e_i = y_i^t - h_s(\mathbf{x}_i^t)$, and $\mathbf{e} = [e_1 \ e_2 \ \cdots \ e_{n_t}]^\top$, then (15) can be simplified as

$$L\left(y_i^t, h_t(\mathbf{x}_i^t)\right) = \left[e_i - \sum_{k=1}^{K} \lambda_k(\mathbf{x}_i^t) z_k \right]^2. \tag{16}$$

Then, we have the minimization problem given as

$$\min_{\mathbf{z}} \frac{1}{2} \sum_{n=1}^{n_t} \left[e_i - \sum_{k=1}^{K} \lambda_k(\mathbf{x}_i^t) z_k \right]^2. \tag{17}$$

As the limited target data may distribute unevenly, a regularization term should be included and the minimization problem is turned to

$$\min_{\mathbf{z}} \frac{1}{2} \sum_{n=1}^{n_t} \left[e_i - \sum_{k=1}^{K} \lambda_k(\mathbf{x}_i^t) z_k \right]^2 + \frac{1}{2} \eta \mathbf{z}^\top \mathbf{z} \tag{18}$$

where η is a parameter to control the importance of the regularization term. For those rules with no target data, the corresponding z_k will be controlled close to zero with the help of η. Therefore, the target model will approach the source model in the space without target data. The LS solution for $\mathbf{z}$ can be computed as

$$\mathbf{z} = \left[\mathbf{W}^\top \mathbf{W} + \eta \mathbf{I} \right]^{-1} \mathbf{W}^\top \mathbf{e} \tag{19}$$

where

$$\mathbf{W} = \begin{bmatrix} \lambda_1(\mathbf{x}_1^t) & \lambda_2(\mathbf{x}_1^t) & \cdots & \lambda_K(\mathbf{x}_1^t) \\ \lambda_1(\mathbf{x}_2^t) & \lambda_2(\mathbf{x}_2^t) & \cdots & \lambda_K(\mathbf{x}_2^t) \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1(\mathbf{x}_{n_t}^t) & \lambda_2(\mathbf{x}_{n_t}^t) & \cdots & \lambda_K(\mathbf{x}_{n_t}^t) \end{bmatrix} \in \mathbb{R}^{n_t \times K}. \tag{20}$$

Note that the solution of bias vector z contains the normalized membership degree $\lambda_k(\mathbf{x}_i^t)$, which is calculated based on the antecedent parameters of the source domain.

2) Computing the Bias With RD: In FS_s, the crisp input $\mathbf{x}_i^t \in \mathbf{X}^t$ is mapped to a fuzzy set. Similarly, we could design a fuzzy set of residuals for the Kth rule as

$$B^k = \left\{ \left(e_i, \mu_{B^k}(e_i)\right) | \mathbf{x}_i^t \in \mathbf{X}^t \right\} \tag{21}$$

where e_i and $\mu_{B^k}(e_i)$ are functions of $\mathbf{x}_i^t$, $e_i = y_i^t - h_s(\mathbf{x}_i^t)$ is the residual of target data, and $\mu_{B^k}(e_i)$ is the membership function. We assume that the target model and the source model share

the same antecedent parameters. Therefore, $\mu_{B^k}(e_i)$, namely, $\mu_{B^k}(\mathbf{x}_i^t)$, can be computed by

$$\mu_{B^k}(\mathbf{x}_i^t) = \prod_{j=1}^{d} \exp\left(\frac{-\left(\mathbf{x}_{i(j)}^t - c_j^k\right)^2}{2\sigma_j^k}\right) \tag{22}$$

where c_j^k and σ_j^k are the shared antecedent parameters as Appendix A. d is the dimension of $\mathbf{x}_i^t$. $\mathbf{x}_{i(j)}^t$ means the jth dimension of the vector $\mathbf{x}_i^t$.

Since the fuzzy set B^k is the residual of the fuzzy rules and the universe of discourse is denoted as $\mathbf{X}^t$, the relationship between the residual of target data e_i and the bias z_k can be established as a defuzzification process. A defuzzifier produces a crisp output of a fuzzy set. Therefore, the crisp output of the fuzzy set B^k is the bias z_k. Using centroid defuzzifier, the final bias z_k can be computed as

$$z_k = \frac{\sum_{i=1}^{n_t} e_i \mu_{B^k}\left(\mathbf{x}_i^t\right)}{\sum_{i=1}^{n_t} \mu_{B^k}\left(\mathbf{x}_i^t\right)}. \tag{23}$$

Note that the solution of bias vector z contains the membership degree $\mu_{B^k}(\mathbf{x}_i^t)$, which is calculated based on the antecedent parameters of the source domain.

3) Bounds of Fuzzy Residual: The ResTL methods that compute the bias with LS and RD are called ResTL_{LS} and ResTL_{RD}, respectively. The bounds of the fuzzy residual $r(\mathbf{x})$ will influence the final performance of the target model learned by ResTL_{LS} and ResTL_{RD}. Actually, there are no strict bounds for the LS fuzzy residual. For those rules with no target data, the corresponding z_k will be controlled close to zero with the help of the regularization term. However, the RD fuzzy residual has strict upper and lower bounds even with sparse target data.

Denote $e_i^t = y_i^t - h_s(\mathbf{x}_i^t)$, $e_{\max}^t = \max[e_1^t, e_2^t, \ldots, e_{n_t}^t]$, and $e_{\min}^t = \min[e_1^t, e_2^t, \ldots, e_{n_t}^t]$. If there is only one target data, then $e_{\max}^t = e_{\min}^t = e_1^t$. Then, (23) can lead to the following inequality:

$$\frac{\sum_{i=1}^{n_t} e_{\min}^t \mu_{B^k}\left(\mathbf{x}_i^t\right)}{\sum_{i=1}^{n_{tL}} \mu_{B^k}\left(\mathbf{x}_i^t\right)} \le z_k \le \frac{\sum_{i=1}^{n_t} e_{\max}^t \mu_{B^k}\left(\mathbf{x}_i^t\right)}{\sum_{i=1}^{n_{tL}} \mu_{B^k}\left(\mathbf{x}_i^t\right)} \tag{24}$$

$$\frac{e_{\min}^t \sum_{i=1}^{n_t} \mu_{B^k}\left(\mathbf{x}_i^t\right)}{\sum_{i=1}^{n_{tL}} \mu_{B^k}\left(\mathbf{x}_i^t\right)} \le z_k \le \frac{e_{\max}^t \sum_{i=1}^{n_t} \mu_{B^k}\left(\mathbf{x}_i^t\right)}{\sum_{i=1}^{n_{tL}} \mu_{B^k}\left(\mathbf{x}_i^t\right)} \tag{25}$$

$$e_{\min}^t \le z_k \le e_{\max}^t. \tag{26}$$

Then, we have

$$\sum_{k=1}^{K} \lambda_k(\mathbf{x}) e_{\min}^t \le r(\mathbf{x}) \le \sum_{k=1}^{K} \lambda_k(\mathbf{x}) e_{\max}^t \tag{27}$$

$$e_{\min}^t \sum_{k=1}^{K} \lambda_k(\mathbf{x}) \le r(\mathbf{x}) \le e_{\max}^t \sum_{k=1}^{K} \lambda_k(\mathbf{x}) \tag{28}$$

where $\lambda_k(\mathbf{x}) = [(\mu^k(\mathbf{x}))/(\sum_{k=1}^{K} \mu^k(\mathbf{x}))]$ as Appendix A, then

$$e_{\min}^t \le r(\mathbf{x}) \le e_{\max}^t. \tag{29}$$

The RD fuzzy residual has strict upper and lower bounds which are the bounds of generalization errors of the target data on the source model.

III. Experiments

In this section, we evaluate the performance of ResTL through experiments on toy examples, three public datasets, and our own pose-dependent tooltip dynamics dataset. The source code for ResTL is available at github.[1]

A. Experimental Setup

1) Methods for Comparison: To provide a more comprehensive study of the proposed method, we evaluate the performance with four state-of-the-art methods for condition shift problem.

Transfer Learning by Boosting (TLB) [28]: The TLB method, called two-stage TrAdaBoost.R2, which is a famous boosting-based regression transfer algorithm, trains the source data and target data together by adaptively adjusting the instance weights.

Residual Approximation (RA) [19]: The RA method builds the target model by computing the offset between the target data and the source model with GP. This method is widely used in condition shift multisource fusion problems, such as the yield of vineyards [19] or surface measurement [41]. Note that the residual function of RA is trained on the target data directly without using the distribution information of the source data.

General Transformation Function (GTF) [42]: GTF is an algorithm-dependent hypothesis transfer learning method, which characterizes the relationship between the source and the target domains by establishing a GTF.

Domain Adaptation Under GeTarS [10]: GeTarS proposes to resample the source instances by reweighting or transforming to reproduce the distribution on the target domains. And the marginal distribution and conditional distribution are embedded to a reproducing kernel Hilbert space (RKHS).

ResTL_{LS}: The proposed ResTL method that computes the bias with the LS method.

ResTL_{RD}: The proposed ResTL method that computes the bias with the RD method.

Target: GP prediction using only the target data.

Source: GP prediction using only the source data.

2) Datasets: In this experiment, we first demonstrate the properties of the proposed ResTL and other methods on toy examples. Then, four real datasets about conditional shift transfer learning problems are adopted for performance comparison.

The *TOOL* dataset consists of tooltip dynamics of three milling tools (T_1, T_2, T_3) of a five-axes CNC machining center for 381 postures [15]. The features are the angle of axes of the machining center. The labels to be predicted are three dynamic parameters at the tooltip: w_n(natural frequency), ξ(damping ratio), and K(stiffness). For each dynamic parameter prediction task, there are six transfer tasks: 1) $T_1 \to T_2$; 2) $T_1 \to T_3$; 3) $T_2 \to T_3$; 4) $T_2 \to T_1$; 5) $T_3 \to T_1$; and 6) $T_3 \to T_2$. Therefore, there are 18 transfer tasks for TOOL datasets.

The *PM2.5*[2] dataset from UCI is the PM2.5 data in five Chinese cities: 1) Beijing; 2) Shanghai; 3) Guangzhou;

[1] https://github.com/gengxiangc/ResTL

[2] http://archive.ics.uci.edu/ml/datasets/Beijing+PM2.5+Data

4) Chengdu; and 5) Shenyang. There is significant compatibility of yearly changes in air conditions due to the characteristics of the city, such as the weather or geographical configuration [21]. However, the PM2.5 data in each city can shift from year to year with the influence of the traffic or economy. Therefore, five transfer learning tasks were constructed from 2014 to 2015 for the five cities.

The *Kin*[3] dataset from Delve aims to predict the distance of the end effector of a robot with 8-D inputs consisting of joint positions, twist angles, and other dynamics parameters. We build two transfer tasks: 1) kin8fm→kin8nm and 2) kin8nm→kin8fm.

The *Bank*[4] dataset from Delve has the objective to predict the rate of rejections of banks. We build two transfer tasks: 1) bank8fm→bank8nm and 2) bank8nm→bank8fm.

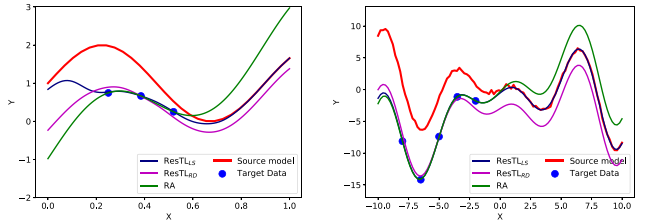
Totally, there are 27 transfer learning tasks. For each task, experiments with a different target data size $n_t = (5\%/10\%/15\%/20\%) \times n_s$ were carried out.

3) Other Settings: The hyperparameters of ResTL, including the number of fuzzy rules K, the spread of the fuzzy set, and others were determined with five-fold cross-validation on the source data. The fuzzy partition method for toy examples and the TOOL dataset was FCM. However, FCM showed poor performance for the high-dimensional dataset, the prototypes would run into the center of gravity of the input dataset. So, a deterministic method Var-Part [45] was adopted for the datasets Kin, Bank, and PM2.5. In addition, the TSK fuzzy system was selected as the base model of ResTL for the PM2.5 dataset whereas GP regressor was selected as that of ResTL for other datasets.

B. Toy Example

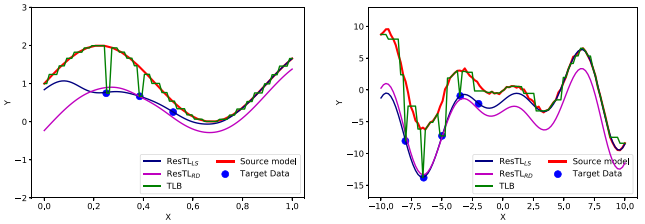
The toy example mainly serves as an illustrative demonstration of the characteristics of ResTL and other methods. The goal is to recover the target model with data from the source model (shown as the red curves in Figs. 2–5, and few target data (blue points in Figs. 2–5). The synthetic curve functions are formulated based on previous researches [10], [19], [28], [42]. For Figs. 2–5(a), the source model is designed as $y = \sin(7x) + 1$ and the target data are drawn from $y = x * \sin(7x) + 0.2$. For Figs. 2–5(b), the source model is designed as $y = \cos(x) * x + \mathcal{N}(0, 0.15)$ and the target data are drawn from $y = \cos(x) * x + x + \mathcal{N}(0, 0.15)$. Then, the characteristics of different algorithms can be explored clearly. More results of toy examples can be found in Appendix B.

1) As shown in Fig. 2(a), for those rules with no target data, the results of ResTL_{LS} almost approach the source model because the corresponding residual is controlled close to zero. However, ResTL_{RD} can only provide global residual with limited target data. The RA method [Fig. 2(b)] can provide similar results in the space where the target data are sufficient, but it will be out of control in other space where the target data are limited.
2) TLB can minimize the training error of both the source data and target data, but the model is easy to overfit when the distribution of target data is far away from the source data as shown in Fig. 3(a) and (b).
3) GTF can provide a good result when the assumption of specific transformation is satisfied [Fig. 4(a)], but the simple function is usually not enough to minimize the training error of target data as shown in Fig. 4(b).
4) GeTarS can match the marginal distribution and conditional distribution in RKHS by reweighting or transforming the samples as shown in Fig. 5(a) and (b). The stability of the distribution matching process suffers from the limited number of the target data.



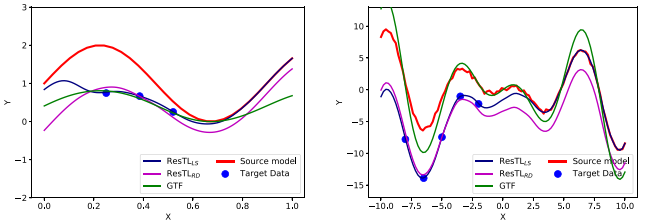
(a) (b)

Fig. 2. ResTL versus RA.



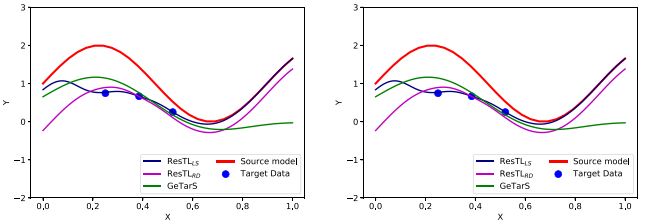
(a) (b)

Fig. 3. ResTL versus TLB.



(a) (b)

Fig. 4. ResTL versus GTF.



(a) (b)

Fig. 5. ResTL versus GeTarS.

[3]http://www.cs.toronto.edu/delve/data/kin/desc.html
[4]http://www.cs.toronto.edu/delve/data/bank/desc.html

TABLE I
RESULTS ON DIFFERENT DATASETS FOR MSE WITH 5% TARGET DATA

Task	n_s/n_t	Target	Source	TLB	GTF	RA	GeTarS	ResTL$_{LS}$	ResTL$_{RD}$
Bank fm→nm	5%	0.004429	0.001186	0.001481	0.003880	0.001172	0.001111	0.001257	**0.001074**
Bank nm→fm	5%	0.015735	0.009813	0.009317	**0.006075**	0.008732	0.006158	0.006993	0.006946
Kin fm→nm	5%	0.072136	0.070270	0.069360	0.062605	0.057401	0.062807	0.053375	**0.049007**
Kin nm→fm	5%	0.002734	0.009788	0.003979	0.001166	0.000818	0.002534	**0.000608**	0.000614
PM2.5 BJ 14→15	5%	1.1536	1.0646	1.3928	0.9859	1.1754	1.0135	1.0226	**0.9664**
PM2.5 CD 14→15	5%	0.7468	0.7950	0.8803	0.7514	0.8266	0.7634	0.7691	**0.7037**
PM2.5 GZ 14→15	5%	0.6629	0.8071	1.1808	0.7267	0.7857	0.7627	0.6577	**0.6068**
PM2.5 SH 14→15	5%	1.0219	1.0623	1.4780	**1.0058**	1.1226	1.0128	1.1558	1.0207
PM2.5 SY 14→15	5%	1.0018	0.9848	1.2473	0.9386	1.1437	**0.9089**	1.2343	0.9266
Tool D10→D6 C	5%	1.2620	0.4490	0.4673	0.4213	**0.3882**	0.4147	0.4203	0.4207
Tool D10→D6 K	5%	1.4315	4.3974	4.2711	4.6640	1.0439	4.0861	1.3854	**0.8137**
Tool D10→D6 W	5%	1.3887	0.3446	0.3436	0.3686	0.2073	0.4483	0.2322	**0.1704**
Tool D10→D8 C	5%	0.3661	0.4908	0.4843	0.3411	0.2421	0.2805	0.2349	**0.1970**
Tool D10→D8 K	5%	0.7992	0.8715	0.9893	0.8022	0.5038	0.6650	0.5231	**0.4783**
Tool D10→D8 W	5%	0.2826	0.5001	0.4548	0.5055	0.0887	0.4705	0.1172	**0.0738**
Tool D6→D10 C	5%	1.2915	0.3964	0.4402	0.3944	0.3387	**0.3013**	0.3444	0.3396
Tool D6→D10 K	5%	3.5467	6.3153	6.1386	7.0441	2.1731	5.9026	1.9971	**1.3235**
Tool D6→D10 W	5%	0.5317	0.2283	0.2493	0.2176	0.1542	0.1591	0.1396	**0.0977**
Tool D6→D8 C	5%	0.3408	0.2354	0.2695	0.1508	0.2452	**0.1336**	0.2821	0.2337
Tool D6→D8 K	5%	1.2629	9.8528	9.2624	9.7339	1.0558	9.0152	1.0870	**0.5979**
Tool D6→D8 W	5%	0.1875	0.1227	0.1523	**0.0760**	0.0921	0.0997	0.1120	0.0918
Tool D8→D10 C	5%	2.2686	0.8101	0.7993	0.8606	0.5464	0.7529	0.5409	**0.4665**
Tool D8→D10 K	5%	2.3106	1.4787	1.6310	1.5567	1.1335	1.4236	1.1088	**1.0502**
Tool D8→D10 W	5%	0.9283	0.5962	0.5751	0.6400	0.2117	0.5399	0.2385	**0.1395**
Tool D8→D6 C	5%	1.9959	0.4893	**0.4839**	0.4894	0.5812	0.5256	0.6329	0.5535
Tool D8→D6 K	5%	2.4587	10.5895	10.5169	11.7349	0.8563	10.2810	2.4517	**0.6885**
Tool D8→D6 W	5%	1.4004	0.2396	0.2296	**0.2046**	0.2226	0.2925	0.2424	0.2200

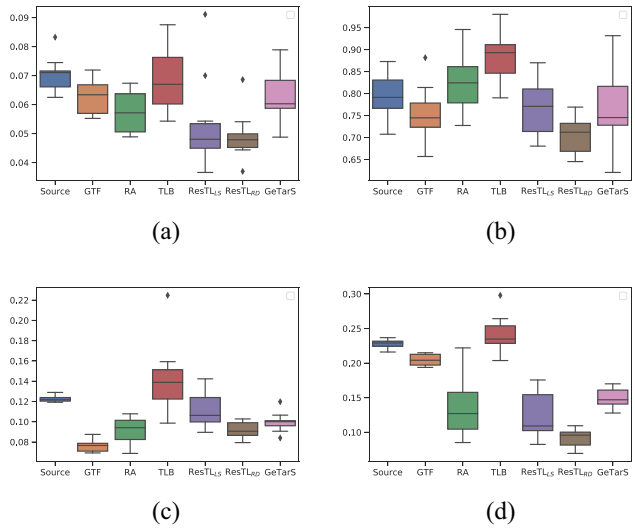


Fig. 6. Boxplots of MSEs with 5% target data. (a) Kin fm to nm. (b) PM2.5 ChengDu. (c) TOOL D6 to D8 W. (d) TOOL D6 to D10 W.

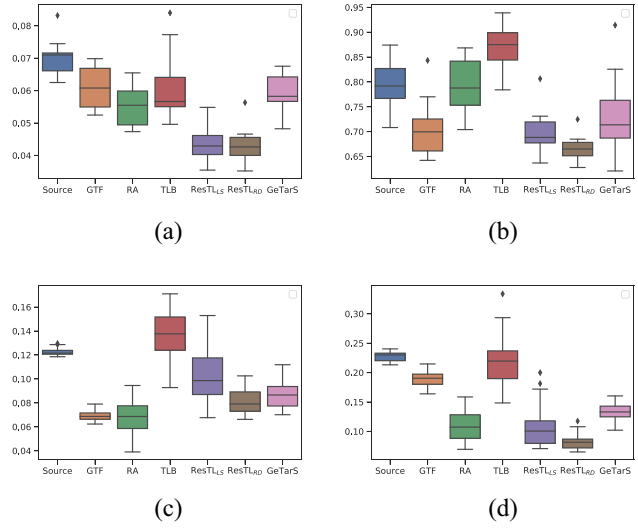


Fig. 7. Boxplots of MSEs with 10% target data. (a) Kin fm to nm. (b) PM2.5 ChengDu. (c) TOOL D6 to D8 W. (d) TOOL D6 to D10 W.

C. Dataset Performance

1) Overall Results: The experimental results of 27 transfer learning tasks are provided in this section to demonstrate the effectiveness of the proposed ResTL. Table I shows the comparisons about the mean-square error (MSE) of the methods with target data size $n_t = 5\% \times n_s$. Other results of $n_t = (10\%/15\%/20\%) \times n_s$ could be found in Appendix B. The number of fuzzy rules K of the experiments was set as: K = 6 for Kin, PM2.5, and Bank, and K = 8 for TOOL. All results are the average values of 20 trials with randomly selected target data. The lowest MSEs of each transfer tasks are marked with bold. From those tables, the following observations can be summarized.

1) Overall, the performance of ResTL$_{\text{RD}}$ is better than ResTL$_{\text{LS}}$ in most tasks. As discussed in Section II, it means that the source and target data in most transfer tasks of this experiment have a global conditional shift, ResTL$_{\text{RD}}$ is more applicable since ResTL$_{\text{LS}}$ will force the target model to approach the source model in the space without target data.
2) ResTL$_{\text{RD}}$ works better than other methods in most tasks when the target data ratio is 5% (18/27 tasks), 10%

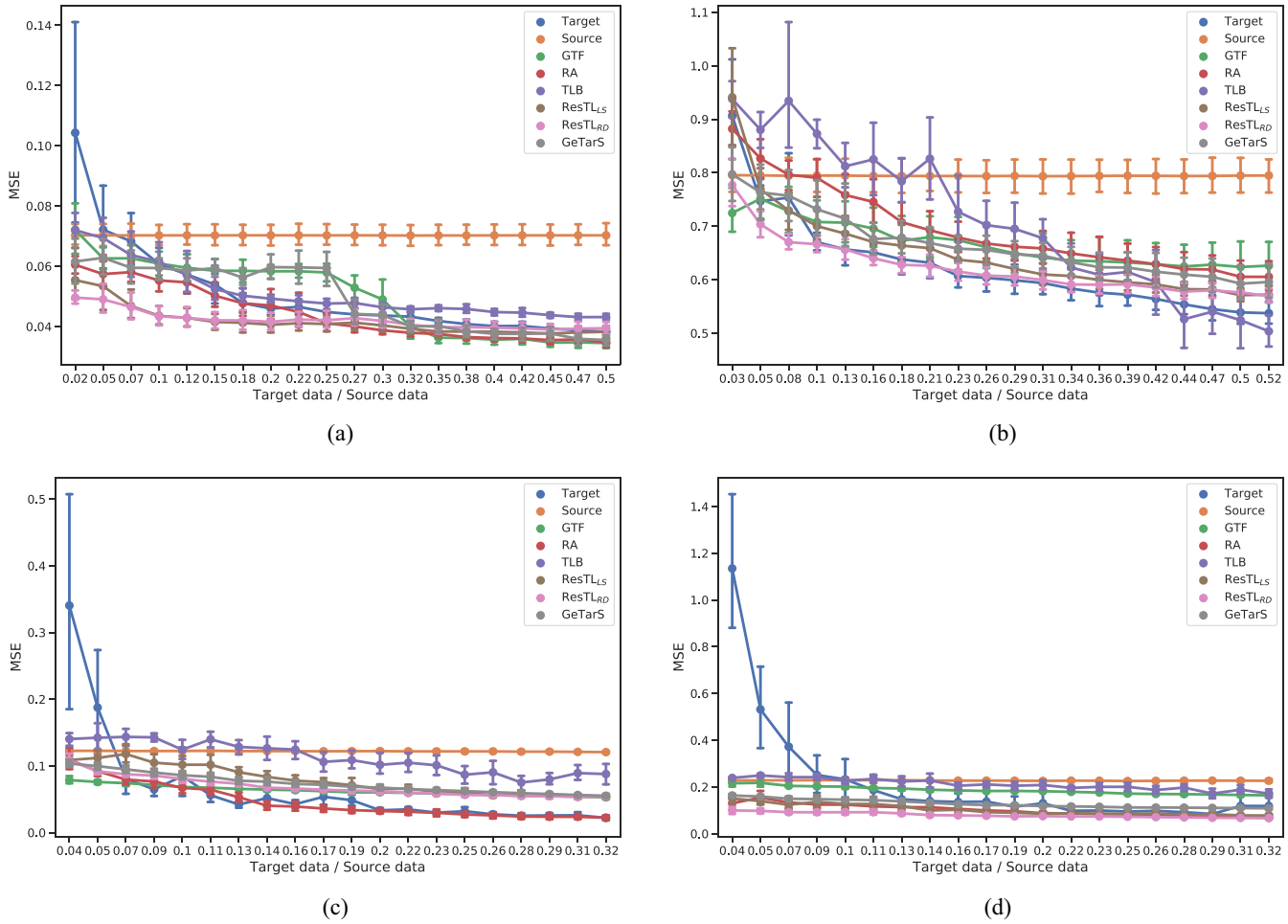


Fig. 8. MSEs with different size of target data. (a) Kin fm to nm. (b) PM2.5 ChengDu. (c) TOOL D6 to D8 W. (d) TOOL D6 to D10 W.

(18/27 tasks), and 15% (16/27 tasks). However, both ResTL_{RD} and ResTL_{LS} show no advantages when the target data size is up to $20\% \times n_s$. The results demonstrate that ResTL is more effective when the target data are not enough.

For further comparison, the boxplots of MSEs are also drawn by 20 trials with random target data (5% and 10%). Some of them are shown in Figs. 6 and 7, and more results can be found in Appendix B. In these figures, the vertical axis refers to the MSEs. It can be found that the proposed ResTL is more effective for most tasks, although RA and GTF show advantages for some specific tasks. In addition, it is obvious that the distribution of the target data has a great influence on the performance of the transfer learning result.

2) Performance on Different Sizes of the Target Data: To further reveal the relationship between the transfer learning performance and the size of the target data, we performed experiments with the size of the target data from 2% up to 50% of n_s. Part of the results is shown in Fig. 8 and others can be found in Appendix B.

For the transfer learning methods applied in the experiment, their MSEs went down as the target data increased, which indicates that more target data are beneficial to enhancing the performance of transfer learning. The MESs of learning with only target data are much larger than that of the transfer learning methods when the size of target data are limited, which means that transfer learning is indispensable in this condition. However, when the target data size increases to a certain level, most of the transfer learning methods were not as good as before and negative transfer may happen. Negative transfer is widely stated as transferring knowledge from the source domain induces a negative impact on the target learner [46], thus it is defined under the target-only baseline. If there is an abundance of the labeled target data, the target-only baseline may be already pretty good. Thus, the knowledge from the source domain may be insignificant and the difference between domains could hurt the generalization.

The influence of the target data size to the transfer learning result of these methods are quite different and complicated. As shown in Fig. 8(a), ResTL_{LS} and ResTL_{RD} show great advantages when the target data are less than 25%, after that, RA and GTF perform better. In Fig. 8(b) and (c), the performance of the target data exceeds all other methods when the target data size is more than 20% of n_s, which means there is a clear boundary for negative transfer in these cases. In Fig. 8(d), ResTL_{RD} and ResTL_{LS} take the leading position with almost all sizes of target data. It can also be found that the performance of ResTL_{RD} is better than ResTL_{LS} for most tasks. As discussed

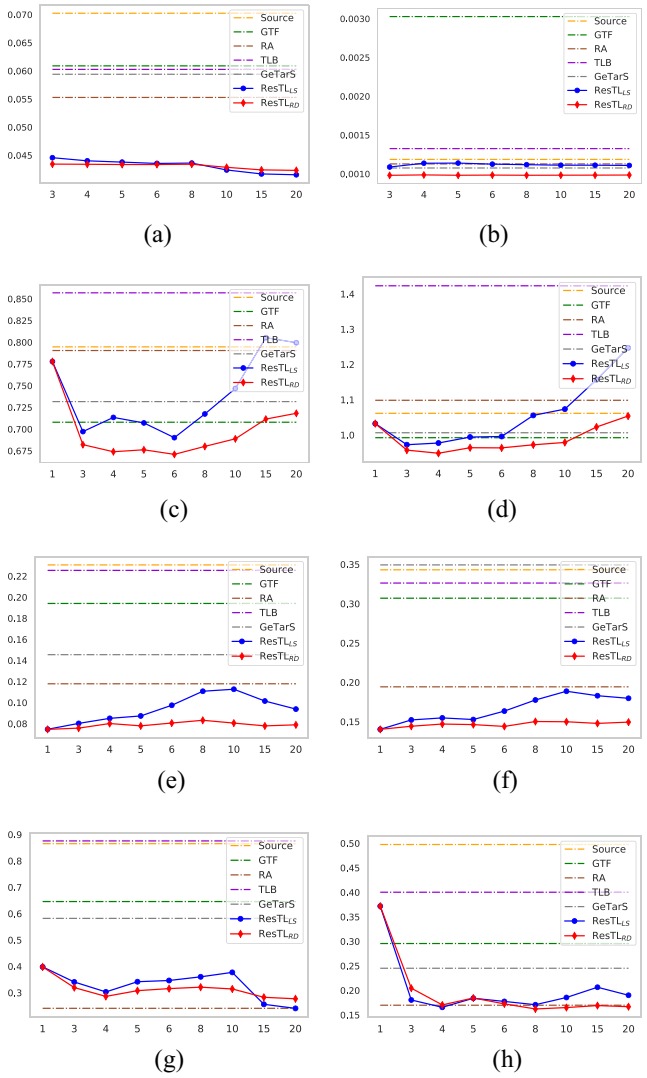


Fig. 9. MSEs with different size of rules. (a) Kin fm to nm. (b) Bank fm to nm. (c) PM2.5 ChengDu. (d) PM2.5 Shanghai. (e) TOOL D6 to D10 W. (f) TOOL D10 to D6 W. (g) TOOL D10 to D8 K. (h) TOOL D10 to D8 C.

in Section II, it means that the source and target data in most transfer tasks of this experiment have a global conditional shift, which is more applicable with ResTL$_{RD}$. In summary, ResTL was much more effective for most tasks in this experiment when the target data were limited.

3) Performance on Different Sizes of Fuzzy Rules: In this section, we will show the performance of ResTL$_{LS}$ and ResTL$_{RD}$ with different sizes of the fuzzy rules. Eight transfer learning tasks were selected and performed with $K = 1, 3, 4, 5, 6, 8, 10, 15, 20$, respectively. In this experiment, the size of the target data was set to 10% of n_s. Fig. 9 shows the MSEs obtained by computing the average of 20 random trials. The experimental result shows that ResTL$_{RD}$ stayed robust with a wide range of the number of fuzzy rules whereas ResTL$_{LS}$ was much more sensitive to K. To guarantee the result of ResTL, we should decide K properly. Actually, the number of fuzzy rules could be selected by cross-validation on the TSK fuzzy model of source data. In summary, the proposed method ResTL$_{RD}$ could stay robust with a wide range of the number of fuzzy rules.

IV. Conclusion

In this article, we proposed a new framework called ResTL that learned the target model for regression problems under a conditional shift situation by preserving the distribution properties of the source data in a model-agnostic way. ResTL can build the target model by learning the residual between the two domains and reusing the antecedent parameters of the source fuzzy system. Furthermore, two algorithms ResTL$_{RD}$ and ResTL$_{LS}$ were introduced for different situations under the proposed framework. The proposed method has been evaluated comprehensively both on the toy example and several real-world datasets, including our own tool-tip dynamics dataset. The results show that the proposed ResTL can provide better performance compared with existing methods when it comes to global conditional shift cases. In addition, we found that the proposed method was more effective when the target data were limited, but the performance was not satisfactory when the target data size further increased. Thus, in the future, we will focus on the negative transfer problem of the proposed method. Moreover, we will extend the proposed framework to multisource cases.

Appendix A
TSK Fuzzy System

The TSK fuzzy system is an intelligent model defined with fuzzy logic, which is essentially a combination of submodels with good interpretability. It is a powerful tool for modeling complex nonlinear systems [43]. Suppose a TSK fuzzy system has d inputs $x_1 \in X_1, \ldots, x_d \in X_d$, one output y $\in Y$, and K rules. Then, the structure of the Kth TSK fuzzy rule for the system consists of IF-THEN in the form

$$\begin{aligned} &\text{IF } x_1 \text{ is } A_1^k \wedge x_2 \text{ is } A_2^k \wedge \ldots \wedge x_d \text{ is } A_d^k \\ &\text{THEN } y \text{ is } f^k(x) k = 1, 2, \ldots, K \end{aligned} \tag{30}$$

where A_j^k is the fuzzy set of the jth input dimension under the kth. A fuzzy set is described by its membership function $\mu_{A_j^k}(x)$ as

$$A_j^k = \left\{ \left(x_j, \mu_{A_j^k}(x_j) \right) | x_j \in X_j \right\}. \tag{31}$$

The shape of the membership function is determined up to the designer. The commonly used membership function is the Gaussian membership function

$$\mu_{A_j^k}(x_j) = \exp\left(\frac{-\left(x_j - c_j^k \right)^2}{2\sigma_j^k} \right) \tag{32}$$

where c_j^k and σ_j^k are the center values and the spread of the fuzzy set A_j^k. These two parameters are defined as the antecedent parameters, which can be calculated by fuzzy partition algorithms such as fuzzy c-means (FCMs).

The membership degree of fuzzy set A^k can be the multiplicative conjunction of membership degrees of each dimension as

$$\mu^k(\mathbf{x}) = \prod_{j=1}^{d} \mu_{A_j^k}(x_j). \tag{33}$$

The normalized membership degree is given by

$$\lambda_k(\mathbf{x}) = \frac{\mu^k(\mathbf{x})}{\sum_{k=1}^{K} \mu^k(\mathbf{x})} \tag{34}$$

where λ_k means the degree of activation of the Kth rule. Then, the output of the TSK fuzzy system can be formulated by a combination of submodels as [44]

$$y = \sum_{k=1}^{K} \lambda_k(\mathbf{x}) f^k(\mathbf{x}). \tag{35}$$

APPENDIX B
SUPPLEMENTARY RESULTS

In the supplementary results, Tables BI–BIII are the overall MSEs on different datasets with $n_t = (10\%/15\%/20\%) \times n_s$, respectively. Figs. B1 and B2 are the boxplots with 5% and 10% target data. Fig. B3 is the comparison results on different sizes of target data. Fig. B4 is the comparisons of different methods on toy examples.

REFERENCES

[1] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.

[2] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[4] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu, "Visual domain adaptation with manifold embedded distribution alignment," in *Proc. 26th ACM Multimedia Conf.*, 2018, pp. 402–410.

[5] J. Li, K. Lu, Z. Huang, L. Zhu, and H. T. Shen, "Transfer independently together: A generalized framework for domain adaptation," *IEEE Trans. Cybern.*, vol. 49, no. 6, pp. 2144–2155, Jun. 2019.

[6] D. Tasche, "Fisher consistency for prior probability shift," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 3338–3369, 2017.

[7] J. Quioñero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset Shift in Machine Learning*. Cambridge, MA, USA: MIT Press, 2009.

[8] J. Huang, A. Gretton, K. Borgwardt, B. Schölkopf, and A. J. Smola, "Correcting sample selection bias by unlabeled data," in *Proc. 19th Int. Conf. Adv. Neural Inf. Process. Syst.*, 2007, pp. 601–608.

[9] S. T. Certo, J. R. Busenbark, H.-S. Woo, and M. Semadeni, "Sample selection bias and heckman models in strategic management research," *Strategic Manag. J.*, vol. 37, no. 13, pp. 2639–2657, 2016.

[10] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang, "Domain adaptation under target and conditional shift," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, pp. 819–827.

[11] M. Pratama, M. de Carvalho, R. Xie, E. Lughofer, and J. Lu, "ATL: Autonomous knowledge transfer from many streaming processes," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manag.*, 2019, pp. 269–278.

[12] A. Haque, Z. Wang, S. Chandra, B. Dong, L. Khan, and K. W. Hamlen, "FUSION: An online method for multistream classification," in *Proc. ACM Conf. Inf. Knowl. Manag.*, 2017, pp. 919–928.

[13] G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean, "Characterizing concept drift," *Data Mining Knowl. Discover.*, vol. 30, no. 4, pp. 964–994, 2016.

[14] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, "Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing," *IEEE Trans. Ind. Informat.*, vol. 15, no. 4, pp. 2416–2425, Apr. 2019.

[15] G. Chen, Y. Li, and X. Liu, "Pose-dependent tool tip dynamics prediction using transfer learning," *Int. J. Mach. Tools Manuf.*, vol. 137, pp. 30–41, Feb. 2019.

[16] J. Munoa *et al.*, "Chatter suppression techniques in metal cutting," *CIRP Ann.*, vol. 65, no. 2, pp. 785–808, 2016.

[17] B. Delhaisse, D. Esteban, L. Rozo, and D. Caldwell, "Transfer learning of shared latent spaces between robots with similar kinematic structure," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, 2017, pp. 4142–4149.

[18] Y. Li, C. Liu, J. Hua, J. Gao, and P. Maropoulos, "A novel method for accurately monitoring and predicting tool wear under varying cutting conditions based on meta-learning," *CIRP Ann.*, vol. 60, no. 1, pp. 487–490, 2019.

[19] X. Wang, T.-K. Huang, and J. Schneider, "Active transfer learning under model shift," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 1305–1313.

[20] J. Garcke and T. Vanck, "Importance weighted inductive transfer learning for regression," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discover. Databases*, 2014, pp. 466–481.

[21] X. Liang, S. Li, S. Zhang, H. Huang, and S. X. Chen, "Pm2.5 data reliability, consistency, and air quality assessment in five Chinese cities," *J. Geophys. Res. Atmos.*, vol. 121, no. 17, pp. 10220–10236, 2016.

[22] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[23] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sydney, NSW, Australia, 2013, pp. 2200–2207.

[24] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan, "Transferable representation learning with deep adaptation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 3071–3085, Dec. 2019.

[25] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, p. 9, 2016.

[26] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. von Bünau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Ann. Inst. Stat. Math.*, vol. 60, no. 4, pp. 699–746, 2008.

[27] W. Dai, Q. Yang, G.-R. Xue, and Y. Yu, "Boosting for transfer learning," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 193–200.

[28] D. Pardoe and P. Stone, "Boosting for regression transfer," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 863–870.

[29] X. Zhang, Y. Zhuang, W. Wang, and W. Pedrycz, "Transfer boosting with synthetic instances for class imbalanced object recognition," *IEEE Trans. Cybern.*, vol. 48, no. 1, pp. 357–370, Jan. 2018.

[30] X. Wang and J. Schneider, "Flexible transfer learning under support and model shift," in *Proc. 27th Int. Conf. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1898–1906.

[31] K. Yu, V. Tresp, and A. Schwaighofer, "Learning Gaussian processes from multiple tasks," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML-05)*, 2005, pp. 1012–1019.

[32] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 759–766.

[33] P. Wei, R. Sagarna, Y. Ke, and Y. S. Ong, "Uncluttered domain subsimilarity modeling for transfer regression," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Singapore, 2018, pp. 1314–1319.

[34] P. Wei, R. Sagarna, Y. Ke, Y.-S. Ong, and C.-K. Goh, "Source-target similarity modelings for multi-source transfer Gaussian process regression," in *Proc. 34th Int. Conf. Mach. Learn. Vol. 70*, 2017, pp. 3722–3731.

[35] Z. Deng, Y. Jiang, K.-S. Choi, F.-L. Chung, and S. Wang, "Knowledge-leverage-based TSK fuzzy system modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 8, pp. 1200–1212, Aug. 2013.

[36] 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 Trans. Fuzzy Syst.*, vol. 24, no. 5, pp. 1079–1094, Oct. 2016.

[37] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Fuzzy regression transfer learning in Takagi–Sugeno fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 6, pp. 1795–1807, Dec. 2017.

[38] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Granular fuzzy regression domain adaptation in Takagi–Sugeno fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 2, pp. 847–858, Apr. 2018.

[39] 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 Trans. Cybern.*, vol. 44, no. 12, pp. 2585–2599, Dec. 2014.

[40] P. Xu, Z. Deng, J. Wang, Q. Zhang, and S. Wang, "Transfer representation learning with TSK fuzzy system," 2019. [Online]. Available: arXiv:1901.02703.

[41] J. Wang, L. Pagani, R. K. Leach, W. Zeng, B. M. Colosimo, and L. Zhou, "Study of weighted fusion methods for the measurement of surface geometry," *Precision Eng.*, vol. 47, pp. 111–121, Jan. 2017.

[42] S. S. Du, J. Koushik, A. Singh, and B. Póczos, "Hypothesis transfer learning via transformation functions," in *Proc. 31st Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 574–584.
[43] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-15, no. 1, pp. 116–132, Jan./Feb. 1985.
[44] M. L. Hadjili and V. Wertz, "Takagi–Sugeno fuzzy modeling incorporating input variables selection," *IEEE Trans. Fuzzy Syst.*, vol. 10, no. 6, pp. 728–742, Dec. 2002.
[45] T. Su and J. G. Dy, "In search of deterministic methods for initializing *K*-means and Gaussian mixture clustering," *Intell. Data Anal.*, vol. 11, no. 4, pp. 319–338, 2007.
[46] Z. Wang, Z. Dai, B. Póczos, and J. Carbonell, "Characterizing and avoiding negative transfer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Long Beach, CA, USA, 2019, pp. 11293–11302.

Gengxiang Chen received the B.S. degree in aerospace manufacturing engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2015, where he is currently pursuing the Ph.D. degree.

His research interest is data-driven intelligent manufacturing.

Yingguang Li (Member, IEEE) received the B.S. and Ph.D. degrees from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1999 and 2004, respectively.

He has been a Professor with the Nanjing University of Aeronautics and Astronautics since 2008. His research interests are primarily focused on data-driven intelligent manufacturing and advanced composite curing technology.

Xu Liu received the M.S. and Ph.D. degrees from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2011 and 2015, respectively.

He is currently working with the School of Mechanical and Power Engineering, Nanjing Tech University, Nanjing. His research interest is data-driven intelligent manufacturing.