Multi-Tree Genetic Programming-based Transformation for Transfer Learning in Symbolic Regression with Highly Incomplete Data

Baligh Al-Helali, Qi Chen, Bing Xue, and Mengjie Zhang,
School of Engineering and Computer Science
Victoria University of Wellington, P.O. Box 600, Wellington 6140, New Zealand
{baligh,qi.chen,bing.xue,mengjie.zhang}@ecs.vuw.ac.nz

***Abstract*—Transfer learning has been considered a key solution for the problem of learning when there is a lack of knowledge in some target domains. Its idea is to benefit from the learning on different (but related in some way) domains that have adequate knowledge and transfer what can improve the learning in the target domains. Although incompleteness is one of the main causes of knowledge shortage in many machine learning real-world tasks, it has received a little effort to be addressed by transfer learning. In particular, to the best of our knowledge, there is no single study to utilize transfer learning for the symbolic regression task when the underlying data are incomplete. The current work addresses this point by presenting a transfer learning method for symbolic regression on data with high ratios of missing values. A multi-tree genetic programming algorithm based feature-based transformation is proposed for transferring data from a complete source domain to a different, incomplete target domain. The experimental work has been conducted on real-world data sets considering different transfer learning scenarios each is determined based on three factors: missingness ratio, domain difference, and task similarity. In most cases, the proposed method achieved positive transductive transfer learning in both homogeneous and heterogeneous domains. Moreover, even with less significant success, the obtained results show the applicability of the proposed approach for inductive transfer learning.**

***Keywords*—Symbolic Regression, Genetic Programming, Incomplete Data, Transfer Learning**

I. Introduction

Data quality is an important factor when developing machine learning algorithms, especially in real-world domains. One of the data quality issues is data incompleteness, which occurs when there are missing values in the used data set. There are three main types of missingness [1]. If the reason of the missingness is related to the missing data itself, it is referred to as missing not at random (MNAR). However, it is missing at random (MAR) when the missingness is related to other observed data. Otherwise, the missingness is called missing completely at random (MCAR). Imputation is one of the widely used approaches to mitigate the missingness issue. It refers to filling in missing data with estimated values producing complete (imputed) data sets [1]. The imputed data can be then used in the learning process in the same way as the complete data can be used.

Symbolic Regression (SR) is the task of constructing a mathematical expression that best fits a given data set [2]. Unlike traditional regression methods, no prior assumption is required in SR. It works by searching for the optimal model and its parameters at the same time. Genetic programming (GP) is a nature-inspired algorithm that generates computer programs for solving a given problem [3]. Based on a high-level definition of the problem, it starts with an initial population of solutions then refines them progressively using variation and selection operators until getting a satisfactory solution [4]. GP does not require any predefined solution, hence, SR tasks have been typically addressed via GP [2].

In SR research, the complete case strategy is mostly used when working on data with missing values. It is done by deleting the incomplete instances then applying the SR learning process on the remaining complete data [5]. This approach has the risk of ignoring some entries that can be important for the learning process, especially when there is a high ratio of incomplete instances. However, a few studies have adopted other approaches. In [6], a method for GP-based SR on incomplete data is presented, where the missing values are handled through prediction models in the evolutionary process. In [7], the missingness is treated as imbalanced data in certain regions of mathematical functions and it is addressed by employing instance weighting schemes. In both [6], [7], the methods are evaluated using synthetic data sets. For SR on real-world incomplete data, a hybrid method combines GP and K-nearest neighbour (KNN) to impute missing values is proposed in [5]. In [8], GP is used for imputation predictor selection and ranking in SR with high-dimensional incomplete data. All these studies are conducted assuming the availability of enough data for the learning process, and no high ratio of missingness has been considered.

Transfer learning aims at extracting the knowledge from a source task and reusing it for learning in a target task [9]. This knowledge can be related to the task domain or to the task itself. The domain $\mathcal{D}$ is defined by two components: a feature space $\mathcal{X}$ and its marginal probability distribution $P(\mathcal{X})$, whereas the task is identified by two components: a label space $\mathcal{Y}$ and a target function $f(.)$. Supervised transfer

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

learning can be categorized into two categories based on the task similarity [9]. It is called inductive transfer learning when the target and source tasks are different, i.e. the difference is related to the label space or the target function, and it is called transductive transfer learning otherwise. Domain-wise, transfer learning can be homogeneous or heterogeneous [10]. Transfer learning is called homogeneous when the feature spaces of the source and target domains are the same, otherwise, it is called heterogeneous. Based on its contribution to the performance of learning in the target domain, knowledge transfer can be positive transfer, neutral transfer, or negative transfer.

Data incompleteness is a challenge when learning from real-world domains. For example, it may cause data inadequacy, which in turn may lead to biased learning models. Although transfer learning is thought of as the state-of-the-art to deal with such situations, it has not been fully investigated for learning from incomplete data. Transfer learning has been intensively investigated for classification and clustering. However, only a few studies have been conducted for transfer learning with SR [11]. Moreover, none of these studies has considered the incompleteness issue.

In this work, the main goal is to develop a GP-based feature transformation method that can positively transfer knowledge from complete source domains to incomplete target domains. The transfer is identified as positive if it improves the SR performance on the target task. To achieve this goal, a multi-tree GP feature construction method is proposed to build a transformation from the source feature space to the target feature space such that the transformed data can help impute the missing values towards improving the SR performance in the target domain. The evaluation aims to examine the applicability of the proposed method in different scenarios according to different levels of lack of knowledge in the target task and its similarity with the source task. For data availability, the target domains are $30\%, 50\%$, and 70% incomplete. For the source-target similarity, both domain-wise types (viz. homogeneous and heterogeneous) are considered along with inductive and transductive task-wise transfer learning.

II. Related Work

One of the first explicit investigations on transfer learning in GP is perhaps conducted in [12]. They proposed transferring knowledge from generations evolved using a source task to a target domain. This knowledge can be the final generation best solution, random subtrees from the final generation, or the best solutions of all generations. This study is extended with further investigation in [13], where the variety of source and target problems is considered. A similar approach is adopted in [14], where common subtrees in the best individuals produced in the source problem are employed for initialization and mutation in the target problem. In [15], a building block selection is proposed as a transfer learning mechanism in GP for SR. However, these studies mostly focus on synthetic tasks in limited domains.

Rather than SR, transfer learning methods in GP are also proposed in different domains such as image classification [16], [17], [18] and arc routing problem [19], [20], [21]. However, the adopted approaches are similar to the previous ones. The main idea is to extract genetic knowledge from the evolutionary process on source domains and use it for seeding the evolutionary process on the target domains. In [22], a different methodology for transfer learning in GP is applied successfully in text classification. Unlike previous works, this work utilizes the knowledge learned from source domains in the target task without reperforming the evolutionary process. The outcomes of these studies might not be able to extend to other domains.

A different approach in transfer learning for SR is presented in [23], [24]. They proposed instance-based transfer learning methods to improve the SR performance in target domains with a few training instances. In [23], a local weighting scheme is utilised to identify the usefulness of source domain instances when used in solving a target task. The weighting of the transferred instances is done based on using differential evolution by searching for optimal weights during the GP evolutionary process in [24]. Both methods show good performance on synthetic and real-world SR problems. However, the methods are not practical when transferring from source domains with a high number of instances.

The most recent and interesting studies are presented in [11], [25], where transfer learning in GP is investigated considering source and target tasks from different domains. They utilised GP-based constructive induction of features for transfer learning, where feature transformations evolved on the source task are transferred to the target task. Such transformations are constructed using multidimensional multiclass GP with multidimensional populations (M3GP [26]). The concluded outcomes of this approach draw several important results encouraging further investigations in this direction. Among the reviewed studies, the adopted GP-based transfer learning approach in [11], [25] is the most similar one to our work here. Their valuable outcomes encouraged us to proceed in our investigations. However, our approach has several important differences that will be presented later in the discussion section, Section V.D.

From this review, it can be seen that no single study has considered transfer learning in GP for SR with incomplete data, and this paper aims to fill this research gap.

III. The Proposed Method

This section is dictated to present the proposed method. It starts with the framework of employing this method for symbolic regression with missing values. After that, the details of the proposed method itself are presented.

A. Multi-tree GP-based Transfer Learning (MTGPTL) for SR with Missing Values

Feature adaptation aims at finding the feature representation that reduces the variation between different domains [10]. One approach to do that is called feature transformation-based adaptation, which learns a projection between source and target domains such that the projected feature distribution

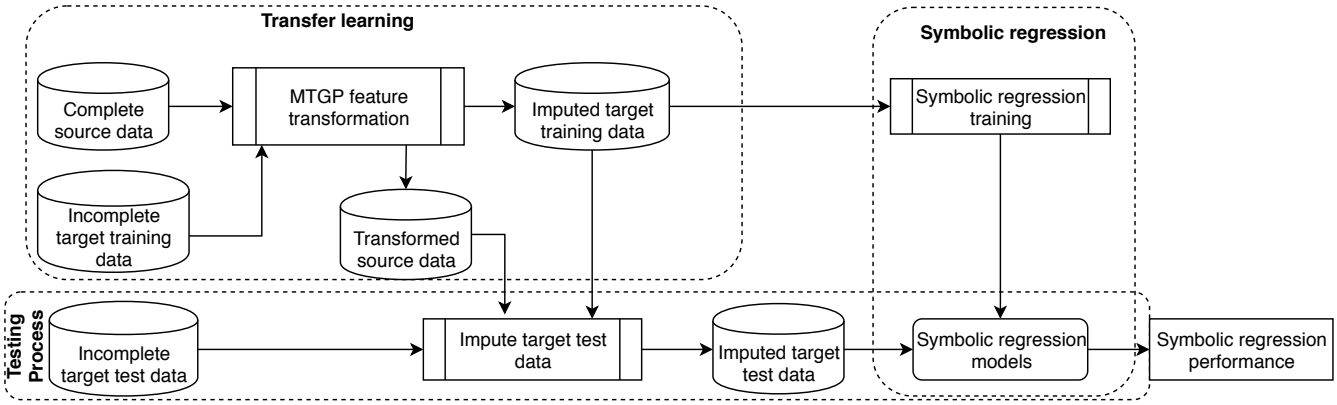

Fig. 1. Multi-tree GP-based transformation for transfer learning in SR with incomplete data.

difference across the two domains can be removed or at least reduced. Such transformations can be symmetric or asymmetric. In the symmetric transformation, both source and target feature spaces are transformed into a common latent feature space. In contrast, one of the feature spaces (target/source) is transformed into the other one (source/target) in the asymmetric transformation [27].

Feature construction is the process of producing a new space of high-level features, $\mathcal{X}_{new}$, from an existing one, $\mathcal{X}_{old}$, based on some criterion, C. Feature construction has been successfully used for improving the performance in several machine learning tasks [28]. It can be used for transfer learning by considering X_{old} to be the feature space of one of the domains and $\mathcal{X}_{new}$ as the feature space of the other domain where the criterion, C, is a metric that measures the transferability of the knowledge extracted by the constructed features.

In this work, the well-known feature construction ability of GP is utilized to present a new approach to transferring knowledge between two domains. It is used to build a transformation between the source features and the target features. This transformation can be then used to transfer data from the source domain to the target domain. It aims to extract knowledge from a complete source domain that can be reused positively for learning in another incomplete target domain. In particular, multi-tree GP (MTGP) [29] is used to construct a new feature space for the source domain by transforming the original features such that this new representation is useful for an SR task on the incomplete target domain. MTGP is chosen for its ability to construct multiple features that can work collectively towards improving the desired task.

The framework of this work is shown in Figure 1. There are three main components in the proposed approach. The first is the transfer learning component. It mainly consists of using MTGP to map the source feature space to the target feature space such that the transformed source data can help impute the target data effectively. The second is the SR process, which is carried out based on the data imputed using the transformed data. Finally, the approach evaluation is done on the unseen target test data.

Although it is designed as a feature-based transformation, what is actually transferred by the proposed method is a set of transformed instances. These instances are obtained by transforming the source data instances using the constructed features. Such a transformation aims to compensate for the lack of knowledge due to the incompleteness in the target data. The transformed instances are used to impute the missing values in the target instances in a manner that improves the corresponding SR performance.

B. Multi-tree GP-based Asymmetric Transformation for Transfer Learning

Multi-tree GP is basically a standard tree-based GP method, where each individual is represented using multiple trees. It is used for constructing multiple features by assigning the expression evolved with each tree as a constructed feature. This mechanism is utilized to find a transformation that maps a source feature space, $\mathcal{X}_S$, to a new feature space, $\bar{\mathcal{X}}_S$, which is compatible with a target feature space, $\mathcal{X}_T$. The proposed method is designed as a wrapper-based method, where the feature transformation construction is evolved based on the regression performance in the target domain. As all individual's trees are evolving simultaneously, the whole set of produced features is constructed such that they perform well together.

The details of the proposed Multi-tree GP transfer learning are given in Algorithm 1. The inputs of the algorithm are N_S-dimensional complete source data, $\mathcal{X}_S^{N_S}$, and incomplete target domain training data with N_T features, $\mathcal{X}_{T,train}^{N_T}$. It then uses MTGP to find an optimal transformation that consists of N_T features constructed from the N_S source features. The transformation is then applied to the source data producing the transformed data, $\bar{\mathcal{X}}_S^{N_T}$, that can be used to impute the target data resulting in complete target data, $\hat{\mathcal{X}}_{T,train}^{N_T}$.

The proposed method is designed in a way that implicitly considers both feature alignment and feature matching. Each individual is represented by a number of trees which is equal

Algorithm 1: Multi-tree GP transformation for incomplete data imputation

Input : Complete source data $\mathcal{X}_S^{N_S}$ with N_S features and incomplete target domain training data $\mathcal{X}_{T,train}^{N_T}$ with N_T features

Output: Complete target domain training data $\hat{\mathcal{X}}_{T,train}^{N_T}$ and transformed source data $\bar{\mathcal{X}}_S^{N_T}$ of N_T features

1 Initialize the population individuals of the first generation, where each one consists of N_T trees;
2 **while** *not stop* **do**
3 **foreach** *individual* I **do**
4 Apply I to get a transformed copy of the source data, i.e. $\bar{\mathcal{X}}_{S,I}^{N_T} = I(\mathcal{X}_S^{N_S})$, where $I : \mathcal{X}^{N_S} \rightarrow \mathcal{X}^{N_T}$;
5 Use the transformed data, $\bar{\mathcal{X}}_S^{N_T}$, to impute the target data, $\mathcal{X}_{T,train}^{N_T}$, using weighted KNN;
6 Compute the fitness value of I to be the regression performance on the imputed data, $fitness_I = WrapperRegressor(\hat{\mathcal{X}}_{T,train,I}^{N_T})$;
7 **end**
8 Select parent(s) from the current generation;
9 Create new individuals (offspring) using genetic operators on the selected parents;
10 Form a new generation from produced offspring and some selected parents;
11 **end**
12 Find the best individual in the last generation, I^*;
13 Return the imputed target data and the transformed source data resulted from applying I^*, i.e. $\hat{\mathcal{X}}_{T,train}^{N_T} = \hat{\mathcal{X}}_{T,train,I^*}^{N_T}$ and $\bar{\mathcal{X}}_S^{N_T} = \bar{\mathcal{X}}_{S,I^*}^{N_T}$;

to the number of the target features. The constructed features are applied to the source instances to produce the transformed data, which are then appended to the target data using the same order of both the target features and the constructed features. The merged data are then used by KNN to impute the missing values. It can be thought of as searching for the solution (individual) whose constructed features (trees) are aligned with the target features such that the performance is improved. This means that the first tree is evolved enhancing the imputability of the first target feature using the first constructed feature. Similar behaviors are true for the remaining features.

KNN imputation replaces each missing value with the weighted average (weighted based on distance) of the k closest instances [30]. The use of weighted KNN (WKNN) imputation can be considered as an implicit weighting operation for the transferred (transformed) instances. However, rather than considering their direct contributions in the regression performance, the source instances are weighted based on their contributions in imputing the target incomplete instances. For measuring the goodness of each individual (transformation), the transformed data are used to impute the incomplete target training data and the SR performance on the imputed data is considered.

There are several approaches for implementing the genetic operators in multi-tree GPs [31]. In this work, the same-index crossover (SIC) approach is used for individuals' crossover. It works by randomly picking a tree index and then applying a standard single-tree GP crossover on the two trees at this index in the two multi-tree individuals. This approach increases the ability of each constructed feature to specialise while improving the performance of being used with the other evolved features. For the mutation, it is performed by applying a standard single-tree GP mutation on a randomly chosen tree. As each individual is intended to be a transformation from the source domain to the target one, the number of trees is set to the number of the features in the target feature space. In addition to random constants, the source problem features form the terminal set of MTGP.

IV. Experiment Setup

The proposed method is compared with two baseline approaches. The first one is traditional learning without using transfer learning denoted as "NoTL". This method uses only the target training data for learning and it is used to show whether the transfer learning is positive. The second approach is learning after transferring the source data without any adaption and this one is denoted as "PlainTL". It simply combines the source domain data and the target domain training data and the combined data are then used for learning the target task. This approach is considered to show the effectiveness of the proposed domain adaption (transformation).

The SR, wrapper regression (step 6 in Algorithm 1), and KNN imputation are carried out using the methods SymbolicRegressor, RandomForestRegressor, and KNNImputer from the python packages gplearn, sklearn, and missingpy, respectively, with default parameters. However, the MTGPTL method is implemented using the DEAP python package. The parameters of MTGPTL are set empirically to max tree depth of 8, crossover rate of 0.8, mutation rate of 0.2, population size of 512, and top-10 elitism.

For evaluation, four real-world data sets are used: Housing, Concrete, Forestfires, and Yacht-hydrodynamics. These data sets are chosen as they represent standard regression problems in the popular machine learning repository UCI [32]. The statistics of the used data sets are shown in Table I. To make the data sets suitable for evaluating transfer learning methods, the data sets are divided into target and source data sets according to some practices in transfer learning as will be described later. For each target data set, 30 incomplete data sets with $30\%, 50\%$, and 70% MAR missingness probabilities are generated. For each experiment, 30 independent GP runs are performed. The target data are split randomly into $70:30$ training: test data sets in each run and all considered methods are evaluated on the same split, then the average performance for each method is computed.

The comparisons between the different methods are carried out based on their impact on the SR performance. Each method

TABLE I
STATISTICS OF THE USED COMPLETE DATA SETS

Data set	#Features	#Instances
Housing	13	507
Concrete	9	1030
Yacht-hydrodynamics (Yacht)	6	308
Forestfires	13	517

is used to impute the incomplete data sets and the imputed data are then used to perform SR. In this work, the relative squared error (RSE) shown in Equation (1) is used to measure the regression error.

$$RSE = \frac{\sum_{i=1}^{n}(y_i - t_i)^2}{\sum_{i=1}^{n}(t_i - \bar{t})^2} \quad (1)$$

where n is the number of instances, y_i is the i^{th} predicted value, t_i is the i^{th} desired value, and $\bar{t}$ is the average of the desired values t_i, $i = 1, 2, 3..., n$.

V. RESULTS AND DISCUSSIONS

A. Homogeneous Transductive Transfer Learning

In this set of experiments, we consider the difference between the source and target tasks to be the marginal distribution, i.e. the two tasks have the same feature space but with different distributions, $P(\mathcal{X}_S) \neq P(\mathcal{X}_T)$. To guarantee this situation, the data sets are modified according to similar practices used in transfer learning studies [33], [34], [23]. Each data set is sorted according to one feature and the first two-thirds of the instances are used as source data and the remaining data are used as target data. Moreover, to make sure that the marginal distributions of the source and the target domains are different, different normal distributions are used to add random values into the target features.

Table II shows the results of homogeneous transfer learning on the considered data sets with different missingness probabilities. For each experiment, the shown results are according to the SR performance on the target domain. The results are obtained from 30 independent runs and the average is shown as "Mean" while "Sdev" is the standard deviation (multiplied by 10 for more readability). The "%" column refers to the missingness ratio and the "ST" column refers to the significance of the difference between the test results of the compared methods based on pair-wise Wilcoxon test with a significance level of 0.05. The symbol "+" ("-") means that the corresponding method outperforms (is outperformed by) the compared method, whereas "=" refers to no significant difference. For each case, an ordered 2-tuple form is used to show the significance indicator of the comparison with the other two methods in the same order they are shown in the table. For example, (=,-) for the NoTL means that the result of using NoTL is similar to those of PlainTL but significantly worse than the results of using MTGPTL.

From the shown results, it is easy to notice the gain brought by using the proposed method. Actually, it achieves positive transfer learning in almost all the considered cases. This is due to the use of MTGPTL transformation, which encourages producing instances that can be used for imputing target incomplete instances positively. That is, the involvement of the transformed source instances led to better SR performance in the target domain. This improvement is more significant in the cases of high ratios of missing values, e.g. 70%. In fact, the higher the missingness ratio, the more superior the proposed method is. Although it resulted in less SR error, MTGPTL has not significantly outperformed the other methods on the Concrete data set when the missingness is 30%. This can be because this data set has comparatively large training data available which may be enough for imputing the missing values without using instances from the source domain.

On the other hand, PlainTL has not achieved any positive results at all. For low missingness ratios, it has no significant impact. This can be because mostly the k neighbors used by WKNN for imputation are from the target domain itself which leads to similar results of using the target alone, i.e. NoTL. However, for high missingness ratios, there is a shortage in the available target data which implies the need to use some source instances in imputing the target incomplete ones. As can be seen, the corresponding results are negative and this is due to the distribution differences between the two domains which are not mitigated when transferring the source data. Such results suggest that transferring knowledge naively might not be successful, or it may even hurt the learning performance in the target domain. In other words, PlainTL does not lead to positive transfer, and, mostly, it causes negative transfer. This means that knowledge should not be transferred plainly which supports the usability of the proposed method.

B. Heterogeneous Transductive Transfer Learning

To conduct experimental work on heterogeneous transductive transfer learning, there is a need for domains with different feature spaces while having the same task. One of the ways to achieve that is by considering subsets of the original features. In this work, the strategy used in [35] is followed to meet the assumptions of the heterogeneous transductive transfer learning. They split the data such that only the first half of the feature space is used in the source domain while the target domain includes all the available features. We refer to this setting as "Half-Full", i.e. a source domain with half of the features and a target one with all features. Moreover, in addition to the "Half-Full" case, the cases "Full-Half" and "Half-Half" are also considered in this work. In "Full-Half", the source domain has all the available features and the target domain has only the half, while "Half-Half" is the extremest situation as each domain has a different half of the features. Note that the first half contains the first 50% of the features and the other half consists of the remaining features.

These settings are applied to the same source and target data sets obtained previously in the homogeneous transfer learning scenario. This makes sure that there is at least a difference in the marginal distributions (in case the extracted feature spaces are equivalent). Although the cases of "Half-Full" and "Full-Half" imply overlapping feature spaces, the two spaces are still

TABLE II
THE SR RESULTS OF HOMOGENEOUS TRANSFER LEARNING ON THE CONSIDERED DATA SETS WITH DIFFERENT MISSINGNESS PROBABILITIES

		NoTL					**PlainTL**					**MTGPTL**				
		Training		**Testing**			**Training**		**Testing**			**Training**		**Testing**		
Data	%	Mean	Sdev	Mean	Sdev	ST	Mean	Sdev	Mean	Sdev	ST	Mean	Sdev	Mean	Sdev	ST
Housing	30	0.4576	0.0613	1.0008	0.1065	(=,-)	0.4576	0.0723	0.9976	0.1497	(=,-)	0.4336	0.0506	**0.965**	0.1194	**(+,+)**
	50	0.7877	0.07	1.0909	0.2298	(+,-)	0.8507	0.0633	1.1411	0.2344	(-,-)	0.7247	0.0499	**1.0109**	0.1429	**(+,+)**
	70	0.8563	0.0606	1.2723	0.2048	(+,-)	0.9562	0.0723	1.3783	0.1587	(-,-)	0.7469	0.0499	**1.1168**	0.1375	**(+,+)**
concret	30	0.9379	0.128	1.1948	0.2574	(=,=)	0.9365	0.1147	1.1951	0.2186	(=,=)	0.9337	0.1043	1.1944	0.2124	(=,=)
	50	1.234	0.1866	1.4817	0.2212	(+,-)	1.3287	0.1416	1.5529	0.2589	(-,-)	1.1436	0.0903	**1.4709**	0.1801	(+,+)
	70	1.2652	0.1325	1.5938	0.2681	(+,-)	1.4128	0.1252	1.7107	0.229	(-,-)	1.1176	0.0839	**1.3724**	0.2285	(+,+)
Forest	30	1.0077	0.0054	1.0339	0.02	(=,-)	1.0095	0.004	1.0357	0.0238	(=,-)	0.9549	0.0035	**0.9748**	0.0196	(+,+)
	50	1.0245	0.0033	1.1136	0.01	(+,-)	1.1099	0.0031	1.1604	0.0378	(-,-)	0.9426	0.0037	**1.0208**	0.0229	(+,+)
	70	1.0261	0.0057	1.1542	0.0539	(+,-)	1.1686	0.0064	1.2427	0.0288	(-,-)	0.8893	0.0038	**1.0195**	0.0202	(+,+)
Yacht	30	0.8753	0.045	1.2331	0.2605	(+,-)	0.8743	0.0334	1.2327	0.2393	(-,-)	0.8232	0.0294	**1.1686**	0.2583	(+,+)
	50	0.9353	0.0362	1.4208	0.5004	(+,-)	1.0102	0.0514	1.489	0.4164	(-,-)	0.8636	0.0285	**1.3071**	0.2549	(+,+)
	70	0.9497	0.0731	1.2699	0.2227	(+,-)	1.0605	0.0468	1.3588	0.3639	(-,-)	0.823	0.0325	**1.0935**	0.2549	(+,+)

different. Note that the "Full-Full" setting implies no change in the two domains and it does not satisfy the heterogeneity assumption. However, it is shown with the same results of the previous section for the sake of easier comparisons.

In this section, MTGPTL is only compared with NoTL. This is because it is not possible to apply PlainTL approach in the heterogeneous situation as feature-based appending between two data sets with different feature spaces is not valid. Table III shows the means of the obtained RSEs for the target test data sets when considering different transfer scenarios on the used data sets. Column "ST" refers to the significance comparison between the proposed method (MTGPTL) and the learning without transfer learning (NoTL), where "+" ("-") means that MTGPTL significantly outperforms (outperformed by) NoTL, whereas "=" means no significant difference. The columns "Full" and "Half" represent the source feature space, $\mathcal{X}_S$, and the rows "Full" and "Half" are for the target one, $\mathcal{X}_T$.

Based on the shown results, the proposed method works well in most of the cases. Moreover, for some data sets, it improved the target task learning process in all cases. An example of this situation is the Forestfires data set. On all the data sets, considering all the features in the source domain, i.e. "Full-*", achieves more positive transfer learning than the other case. This can be due to the inclusion of the target features in the source feature space. Although they still have different distributions, the constructed features might be more capable of reducing this difference than mapping totally independent features.

C. Heterogeneous Inductive Transfer Learning

The last considered scenario for evaluating the proposed method is transfer knowledge where the two tasks are different and they have different domains as well. For this purpose, several transfer learning experiments are conducted considering all possible combinations of employing the source domain from one task to solve the target task of another one. The obtained results from this scenario are given in Table IV. It shows the RSEs average of the SR performance on the row-wise target tasks considering the source of the task in the corresponding columns.

TABLE III
THE SR TEST RESULTS ON EACH DATA SET WITH DIFFERENT FEATURE SPACE SETTINGS

		$\mathcal{X}_S$	**Full**			**Half**		
Data	$\mathcal{X}_T$	%	NoTL	MTGPTL	ST	NoTL	MTGPTL	ST
Housing	Full	30	1.0008	**0.965**	+	1.0776	1.0246	=
		50	1.0909	**1.0109**	+	1.1938	**1.1425**	+
		70	1.2723	**1.1168**	+	1.3289	**1.5722**	+
	Half	30	1.1771	**1.0725**	+	**1.1384**	1.1571	-
		50	1.2178	1.1414	=	1.2845	1.2949	=
		70	1.2567	1.2715	=	1.3256	**1.3526**	+
Concrete	Full	30	1.1948	1.1944	=	1.1937	**1.1574**	+
		50	1.4817	**1.4709**	+	**1.385**	1.9755	-
		70	1.5938	**1.3724**	+	1.5062	**1.4262**	+
	Half	30	1.3996	**1.1996**	+	1.3289	**1.1937**	+
		50	1.4663	1.4856	=	1.5363	**1.1411**	+
		70	1.8341	**1.6735**	+	1.728	**1.6351**	+
Forest	Full	30	1.0339	**0.9748**	+	1.1518	**1.1042**	+
		50	1.1136	**1.0208**	+	1.1532	**1.0527**	+
		70	1.1542	**1.0195**	+	1.1326	**1.0749**	+
	Half	30	1.1413	**1.1007**	+	1.1275	**1.1466**	+
		50	1.1626	**1.1432**	+	1.1529	**1.1413**	+
		70	1.1644	**1.1374**	+	1.1622	**1.1567**	+
Yacht	Full	30	1.2331	**1.1686**	+	0.9911	**0.9435**	+
		50	1.4208	**1.3071**	+	1.2524	**1.0598**	+
		70	1.2699	**1.0935**	+	1.3589	**1.1548**	+
	Half	30	**1.6441**	1.678	-	1.8554	**1.8164**	+
		50	2.1265	2.2286	=	2.0245	2.0971	=
		70	2.3298	**2.1717**	+	2.2492	**2.1348**	+

The obtained results show the applicability of the MTGPTL method to transfer learning between different tasks from different domains, however, they also show that this transfer learning is task-dependent. That is, some tasks served well as source tasks while other tasks are better as target tasks. In fact, it can be noticed the case where a specific task, when used as a target task, can benefit improving the SR in other target tasks but it receives no gain as a target task. Such a task can be called a "giver" task. For example, when the Concrete problem used as a source task by MTGPTL, most of the considered target tasks got improved SR performance. This can be noticed from the ST column under the Concrete header. However, in return, when considered as a target task by MTGPTL, with the exception of the Concrete task itself, no other source task could make any improvement.

TABLE IV
THE SR TEST RESULTS OF TRANSFER LEARNING BETWEEN DIFFERENT DATA SETS WITH DIFFERENT MISSINGNESS PROBABILITIES

		Housing			**Concrete**			**Forest**			**Yacht**		
	%	NoTL	MTGPTL	ST	NoTL	MTGPTL	ST	NoTL	MTGPTL	ST	NoTL	MTGPTL	ST
Housing	30	1.0008	**0.965**	+	1.0654	**0.9889**	+	**1.1737**	1.2658	-	**1.551**	1.647	-
	50	1.0909	**1.0109**	+	**0.8941**	0.9632	-	0.9183	0.9104	=	**0.8833**	1.4831	-
	70	1.2723	**1.1168**	+	**1.1612**	1.1884	-	2.8755	**1.0831**	+	**1.0226**	1.1196	-
Concrete	30	**1.0679**	1.3881	-	1.1948	1.1944	=	1.5289	1.5175	=	**1.0877**	1.2624	-
	50	**1.2399**	3.4974	-	1.4817	1.4709	=	**1.5172**	3.1503	-	**1.339**	3.4476	-
	70	**1.7452**	2.1219	-	1.5938	**1.3724**	+	**1.5234**	4.6791	-	**1.6644**	1.9795	-
Forest	30	1.122	1.1205	=	1.1223	**1.1146**	+	1.0339	**0.9748**	+	1.1218	**1.1069**	+
	50	1.1337	**1.0453**	+	1.1207	**1.0883**	+	1.1136	**1.0208**	+	1.1194	**1.0791**	+
	70	1.1193	**0.8702**	+	1.1354	**0.8813**	+	1.1542	**1.0195**	+	1.134	**0.8978**	+
Yacht	30	1.2569	**1.008**	+	1.2154	**1.0325**	+	1.2574	**1.0936**	+	1.2331	**1.1686**	+
	50	**1.3137**	1.4262	-	1.3227	**1.2677**	+	**1.3909**	2.4946	-	1.4208	**1.3071**	+
	70	**1.1521**	1.1979	-	**1.114**	1.1993	-	**1.1495**	1.5274	-	1.2699	**1.0935**	+

On the other hand, when considering the Forestfires task as a target task, it benefits from the use of any source task in almost all missingness cases. However, it is not useful as a source task for any other target task. In contrast to the "giver" task, this one can be called a "receiver" task. These results indicate that the transferability between different tasks may not be symmetric. Such a conclusion can be due to the asymmetric nature of the proposed method.

D. Further Discussions

One possible question in the case of homogeneous transfer learning would be: as both domains have the same feature space, why not constructing one-to-one transformation mapping the corresponding features in the two domains to each other instead of considering the other features? Actually, the answer to this question is related to the nature of transfer learning in general. As there is a marginal distribution difference, target domain features occur at different rates compared to the source domain. This distortion is called a covariate shift. Moreover, symmetric features may have different meanings. This situation is called feature bias [36].

For the wrapper regression method used to evaluate the transformed knowledge in the transfer learning, the use of SR led to a dramatic improvement in the performance of the proposed method, however, we sacrificed it preferring less computational complexity by using a random forest regressor with default settings. Such a consideration is also useful to avoid the bias caused by adapting the learning process towards the desired SR task.

As mentioned above, the most similar studies to our work are [11], [25]. Here, we point out the main aspects in which our approach is different. First of all, the incompleteness issue, which is an essential part of our work, is not considered in their works. Actually, in [11], the importance of the incompleteness is mentioned referring to its possible impact on the transfer learning, but it is beyond the scope of their work. Moreover, while it is not the main concern in their works, we double considered the lack of knowledge, which is the main motivation behind transfer learning. In our work, only one-third of the data is used as a target data set and, on top of that, different missingness ratios are imposed on this data set.

Secondly, from a transfer learning perspective, what is transferred by their approach is the constructed feature transformation learned on the source domain to be reused for building learning models on the target domain. That is, no instance-based knowledge is transferred as only the way of constructing features is transferred from the source domain to the target domain. However, for our approach, the transferred knowledge is a set of transformed instances. Regarding the process of extracting such knowledge, their transformations are constructed based on improving the performance on the source task and the target data are not involved in the evolutionary process. Consequentially, some issues like feature alignment between source and target feature spaces required to be treated carefully before transfer learning. In contrast, our transformations are constructed depending on improving the missing values imputation leading to better performance on the target task. No explicit feature alignment is required by our method as the feature transformation is constructed while enforcing a temporal match between the constructed features and the target features.

VI. CONCLUSIONS AND FUTURE WORK

In this study, a GP-based transfer learning method is proposed to utilise knowledge learned from complete domains for learning on different, but related, domains that suffer from the incompleteness problem. It is based on constructing multi-tree GP feature transformations that transfer data from a complete source domain to improve handling the missing values in an incomplete target domain. The proposed method performs feature alignment between the constructed features and the target domain features while increasing the transferability of the overall transformation. This method is applied to SR with different ratios of missing values considering various transfer learning scenarios.

The obtained experimental results are encouraging in both homogeneous and heterogeneous domains. For transductive transfer learning, MTGPTL is shown to be able to positively transfer knowledge in almost all the considered cases. This indicates its ability to address the domain-wise variability between similar tasks. This is true when the difference is in the marginal distribution as it is true for tasks with different

feature spaces as long as the task is the same in the two domains. On the other hand, MTGPTL is less successful when the difference is in the task itself. For heterogeneous inductive transfer learning, the results vary according to the considered source-target tasks.

One of the important conclusions of this work is that the GP feature construction ability can be utilized successfully for mapping a task domain to another different one in order to bridge the difference between the two domains. Another conclusion is the potential improvement that transfer learning could bring when employed for addressing the data incompleteness issue. However, this method does not seem to be suitable for high-dimensional data. For future work, this method can be adapted for other machine learning tasks, e.g. classification. The process of bridging the two domains by this method based on implicit matching could be improved by considering some distribution similarity measures. Moreover, there is a need for addressing the time complexity issue which represents the main limitation of the proposed method.

REFERENCES

[1] A. R. T. Donders, G. J. Van Der Heijden, T. Stijnen, and K. G. Moons, "A gentle introduction to imputation of missing values," *Journal of clinical epidemiology*, vol. 59, no. 10, pp. 1087–1091, 2006.

[2] J. R. Koza, "Genetic programming as a means for programming computers by natural selection," *Statistics and computing*, vol. 4, no. 2, pp. 87–112, 1994.

[3] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic programming: an introduction*. Morgan Kaufmann San Francisco, 1998, vol. 1.

[4] N. F. McPhee, R. Poli, and W. B. Langdon, "Field guide to genetic programming," 2008.

[5] B. Al-Helali, Q. Chen, B. Xue, and M. Zhang, "A hybrid GP-KNN imputation for symbolic regression with missing values," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2018, pp. 345–357.

[6] T. Brandejsky, "Model identification from incomplete data set describing state variable subset only–the problem of optimizing and predicting heuristic incorporation into evolutionary system," in *Nostradamus 2013: Prediction, Modeling and Analysis of Complex Systems*. Springer, 2013, pp. 181–189.

[7] E. Vladislavleva, G. Smits, and D. Den Hertog, "On the importance of data balancing for symbolic regression," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 2, pp. 252–277, 2010.

[8] B. Al-Helali, Q. Chen, B. Xue, and M. Zhang, "Genetic programming for imputation predictor selection and ranking in symbolic regression with high-dimensional incomplete data," in *Australasian Joint Conference on Artificial Intelligence*. Springer, 2019, pp. 523–535.

[9] S. J. Pan, Q. Yang, *et al.*, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[10] L. Zhang, "Transfer adaptation learning: A decade survey," *arXiv preprint arXiv:1903.04687*, 2019.

[11] L. Muñoz, L. Trujillo, and S. Silva, "Transfer learning in constructive induction with genetic programming," *Genetic Programming and Evolvable Machines*, pp. 1–41, 2019.

[12] T. T. H. Dinh, T. H. Chu, and Q. U. Nguyen, "Transfer learning in genetic programming," in *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2015, pp. 1145–1151.

[13] E. Haslam, B. Xue, and M. Zhang, "Further investigation on genetic programming with transfer learning for symbolic regression," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016, pp. 3598–3605.

[14] D. O'Neill, H. Al-Sahaf, B. Xue, and M. Zhang, "Common subtrees in related problems: A novel transfer learning approach for genetic programming," in *Evolutionary Computation (CEC), 2017 IEEE Congress on*. IEEE, 2017, pp. 1287–1294.

[15] B. Muller, H. Al-Sahaf, B. Xue, and M. Zhang, "Transfer learning: a building block selection mechanism in genetic programming for symbolic regression," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2019, pp. 350–351.

[16] M. Iqbal, B. Xue, H. Al-Sahaf, and M. Zhang, "Cross-domain reuse of extracted knowledge in genetic programming for image classification," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 569–587, 2017.

[17] M. Iqbal, B. Xue, and M. Zhang, "Reusing extracted knowledge in genetic programming to solve complex texture image classification problems," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2016, pp. 117–129.

[18] M. Iqbal, H. Al-Sahaf, B. Xue, and M. Zhang, "Genetic programming with transfer learning for texture image classification," *Soft Computing*, pp. 1–13, 2019.

[19] M. A. Ardeh, Y. Mei, and M. Zhang, "A novel genetic programming algorithm with knowledge transfer for uncertain capacitated arc routing problem," in *Pacific Rim International Conference on Artificial Intelligence*. Springer, 2019, pp. 196–200.

[20] ——, "Genetic programming hyper-heuristic with knowledge transfer for uncertain capacitated arc routing problem," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2019, pp. 334–335.

[21] ——, "Transfer learning in genetic programming hyper-heuristic for solving uncertain capacitated arc routing problem," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019, pp. 49–56.

[22] W. Fu, B. Xue, M. Zhang, and X. Gao, "Transductive transfer learning in genetic programming for document classification," in *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer, 2017, pp. 556–568.

[23] Q. Chen, B. Xue, and M. Zhang, "Instance based transfer learning for genetic programming for symbolic regression," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019, pp. 3006–3013.

[24] ——, "Differential evolution for instance based transfer learning in genetic programming for symbolic regression," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2019, pp. 161–162.

[25] L. Trujillo, L. Muñoz, U. López, and D. E. Hernández, "Untapped potential of genetic programming: Transfer learning and outlier removal," in *Genetic Programming Theory and Practice XVI*. Springer, 2019, pp. 193–207.

[26] L. Munoz, S. Silva, and L. Trujillo, "M3gp–multiclass classification with gp," in *European Conference on Genetic Programming*. Springer, 2015, pp. 78–91.

[27] M. Friedjungová and M. Jirina, "Asymmetric heterogeneous transfer learning: A survey." in *DATA*, 2017, pp. 17–27.

[28] B. Xue and M. Zhang, "Evolutionary feature manipulation in data mining/big data," *ACM SIGEVOlution*, vol. 10, no. 1, pp. 4–11, 2017.

[29] A. Lensen, B. Xue, and M. Zhang, "Generating redundant features with unsupervised multi-tree genetic programming," in *European Conference on Genetic Programming*. Springer, 2018, pp. 84–100.

[30] L. Beretta and A. Santaniello, "Nearest neighbor imputation algorithms: a critical evaluation," *BMC medical informatics and decision making*, vol. 16, no. 3, p. 74, 2016.

[31] A. Lensen, B. Xue, and M. Zhang, "Genetic programming for evolving similarity functions for clustering: Representations and analysis," *Evolutionary computation*, pp. 1–29, 2019.

[32] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[33] H. Zuo, G. Zhang, W. Pedrycz, V. Behbood, and J. Lu, "Fuzzy regression transfer learning in takagi–sugeno fuzzy models," *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 6, pp. 1795–1807, 2016.

[34] D. Pardoe and P. Stone, "Boosting for regression transfer," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Omnipress, 2010, pp. 863–870.

[35] P. Zhao, S. C. Hoi, J. Wang, and B. Li, "Online transfer learning," *Artificial Intelligence*, vol. 216, pp. 76–102, 2014.

[36] K. R. Weiss and T. M. Khoshgoftaar, "An investigation of transfer learning and traditional machine learning algorithms," in *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2016, pp. 283–290.