When and How to Transfer Knowledge in Dynamic Multi-Objective Optimisation

Minku, Leandro; Ruan, Gan; Menzel, Stefan; Sendhoff, Bernhard; Yao, Xin

When and How to Transfer Knowledge in Dynamic Multi-objective Optimization

Gan Ruan[§], Leandro L. Minku[§], Stefan Menzel[†], Bernhard Sendhoff[†] and Xin Yao[§‡]
[§] CERCIA, School of Computer Science, University of Birmingham, UK
[†] Honda Research Institute Europe GmbH, Offenbach 63073, Germany
[‡] Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China.
{GXR847, L.L.Minku, X.Yao}@cs.bham.ac.uk, {stefan.menzel, bernhard.sendhoff}@honda-ri.de

***Abstract*—Transfer learning has been used for solving multiple optimization and dynamic multi-objective optimization problems, since transfer learning is able to transfer useful information from one problem to help solving another related problem. This paper aims to investigate when and how transfer learning works or fails in dynamic multi-objective optimization. Through computational analyses on a number of dynamic bi- and tri-objective benchmark problems, we show that transfer learning fails on problems with fixed Pareto optimal solution sets and under small environmental changes. We also show that the Gaussian kernel function used in the existing transfer learning-based method is not always adequate. Therefore, transfer learning should be avoided when dealing with problems for which transfer learning fails and other kernel functions should be used when the Gaussian kernel is inadequate. This paper proposes novel strategies and kernel functions that can be used in such cases. Experimental studies have demonstrated the superiority of our proposed techniques to state-of-the-art methods, on a number of dynamic bi- and tri-objective test problems.**

***Keywords*—Evolutionary Algorithms; Transfer Learning; Dynamic Multi-objective Optimization; Prediction-based Method**

I. Introduction

Transfer learning [1] is a kind of machine learning method that is able to transfer the knowledge from a source task to a target task. This inherent characteristic of transfer learning makes it intuitive to apply transfer learning to explore useful experience that have been obtained in one task/problem to solve another related task/problem. The reason is that some similar or related problems/tasks may share some common features, which help to transfer experience from one problem to help solving another problem. As a result, computational resources can be significantly saved when solving similar problems later.

The first attempt at the application of transfer learning to evolutionary computation is by Gupta et. al [2] on a framework of evolutionary multi-tasking optimization (EMT). Subsequently, additional work around transfer optimization for EMT has been proposed. For instance, Da et. al [3] designed an online learning method to seamlessly curb the negative influence of transfer learning in EMT.

Although transfer learning has recently achieved some successes in the field of EMT, it has seldomly been studied in evolutionary multi-objective optimization (EMO). Generally speaking, dynamic multi-objective optimization problems (DMOPs) [4] are a kind of multi-objective optimization problems which involve a series of problems whose objectives change over time As we normally assume that the changing problems are related, there are good opportunities for the application of transfer learning in dynamic multi-objective optimization (DMO). However, there has been so far only one published paper on DMO, introducing transfer learning-based dynamic multi-objective optimization algorithms (Tr-DMOEAs) [5].

The key challenge in DMO is how to constantly trace a changing Pareto optimal front (POF) and/or Pareto optimal set (POS) before the next environment change [6]. Aiming at this goal, researchers have proposed a prediction-based method [6] [7]. This kind of method predicts what the good solutions in the next environment are after learning the regularity of the environment changes. In most prediction-based approaches, it is implicitly assumed that the evolution of the solutions used to train and test the prediction model obeys a fixed independent and identical probability distribution. However, this is not always true under dynamic environments in optimization, since the environmental changes may result in different evolution patterns over time. Consequently, the prediction model based on the incorrect assumption may cause inaccurate prediction of optimal solutions. Transfer learning, which does not make this assumption, is a good candidate for solving DMOPs if it can learn and exploit the relationship among different problems.

The main idea behind Tr-DMOEA is to transfer solutions in the Pareto front (POF) of the previous environment to generate an initial population for the next environment. Experimental studies [5] have shown the superiority of Tr-DMOEAs over the state-of-the-art in DMO. However, the results also showed that Tr-DMOEA does not always work well. It is therefore important to understand why and when transfer learning does not work well. Only after understanding it can we make some improvements regarding transfer learning in DMO.

Considering the general process of transfer learning [1], there are three main components: (1) what to transfer; (2) when to transfer; (3) how to transfer. In terms of the first question, good solutions found for the previous environment are typically aimed to be transferred to the next environment in DMO. However, there are no straightforward answers to the other two questions. This paper thus performs an investigation to answer these two questions, and proposes a new method for

transfer learning in DMO, which overcomes key weaknesses of the existing method [5].

The main contributions of this paper are as follows:

1) A comprehensive computational investigation into transfer learning to find out when transfer learning does and does not work well in DMO. We find that transfer learning does work well on problems with fixed POS and under small environmental changes.
2) A mathematical proof that the Gaussian kernel function in Tr-DMOEA is not ideal; we show that a linear kernel function overcomes the key weakness of the Gaussian kernel function.
3) An improved method of transfer learning in DMO that combines copied solutions from the previous environment and transferred solutions through linear kernels as the initial population in the new environment. Experimental results have shown that our proposed method is better than the state-of-the-art methods in most cases.

The rest of this paper is organized as follows: Section II answers the question when transfer learning works/fails in DMO and proposes to avoid transfer learning when it fails. In Section III, we present how transfer learning works in DMO and propose to replace the Gaussian kernel function with a linear one. Experimental studies of the proposal and comparison with the existing methods are given in Section IV. Section V states the summary and some potential future work.

II. When Does Transfer Learning Work/Fail in DMO?

This section analyzes when transfer learning works or fails in DMO, such that some potential improvements can be put forward.

A. Computational Investigation into Transfer Learning in DMO

Generally, whenever a new environment change happens, there is a population that has already been optimized to the previous environment. To check whether transfer learning works, we compare the quality of the following solutions on the new environment: (1) transferred solutions and (2) solutions copied from the previously optimized population. For transfer learning to be considered as successful, (1) should be of at least similar (and ideally better) quality than (2) on the new environment.

The IEEE CEC 2015 Benchmark problems [8] are selected as the test problems, which have 12 bi- and tri-objective problems with different features. For the parameters of these problems, there are 20 changes. In order to study the effectiveness of Tr-DMOEAs in different dynamics, there are three dynamics with different severity of change (i.e., n_t = 10, 1 and 20). They represent the environment changes are medium, large and small, respectively. Within each change, the population is forced to run 50 generations (i.e., τ_t = 50), which enables the population to converge. This corresponds to the parameter settings of C4, C6 and C8 in Table I. At the beginning of the algorithm, the population iterates for 50 generations, which also enables the population to converge.

The experimental setup is as follows:

1) The population size is set as 200, as in previous work [5];
2) RMMEDA [9] as the optimization algorithm. RMMEDA is a regularity model-based multi-objective estimation of distribution algorithm. It is able to make the population converge quickly before the next change, avoiding the cases that unconverged solutions affect the results. The state-of-the-art Tr-DMOEA [5] is adopted.
3) In DMO, the key point is to find the optimal solutions as soon as possible before next change. In this case, the better the generated solutions after each change, the better the Tr-DMOEA. Therefore, if transferred solutions are better than copied solutions from the previous environment, we consider that transfer learning works well. In the contrary, if transferred solutions are worse than copied solutions, transfer learning fails.

Inverted Generational Distance (IGD) [10] is used to compare the performance of two solutions sets. IGD can measure the diversity and convergence of a solution set found by an algorithm, so it can give us a comprehensive understanding about the performance of compared algorithms. MIGD [11] is a modified version of IGD, which is the average IGD values in all changes. The smaller the MIGD the better the algorithm.

TABLE I
Configurations of benchmark function parameters.

	n_t	τ_t.	τ_T
C1	10	5	150
C2	10	10	250
C3	10	25	550
C4	10	50	1050
C5	1	10	250
C6	1	50	1050
C7	20	10	250
C8	20	50	1050

n_t, τ_t and τ_T are the severity of change, frequency of change and maximum number of iterations, respectively.

Here, MIGD values for Tr-RMMEDA and RMMEDA are compared. After each environmental change, RMMEDA gets an initial population found in last generation of the previous environment, while Tr-RMMEDA obtains an initial population through transfer learning. These two initial populations are evaluated in the new environment and then used to calculate the IGD values. MIGD values for Tr-RMMEDA and RMMEDA are the average of IGDs under 20 environments. The comparison results are shown in Table II, in which $Tr.$ and $Copy$ mean MIGD values for Tr-RMMEDA and RMMEDA respectively.

It can be observed from the table that transferred solutions are all worse than those from the previous environment on problems HE2, HE7 and HE9, no matter what kinds of changes are present in these benchmark problems. All HE problems have fixed POS. Therefore, it can be concluded that transfer learning fails on problems with fixed POS. For other problems, transferred solutions are better than those copied from the previous environment when n_t = 1 (C6). Regarding other two

TABLE II
MIGD VALUES FOR TR-RMMEDA AND RMMEDA.

Prob.	C4		C6		C8	
Methods	Tr.	Copy	Tr.	Copy	Tr.	Copy
FDA4	**0.082**	0.129	**0.098**	3.987	0.081	0.081
FDA5	0.389	**0.179**	**1.294**	5.211	0.325	**0.115**
FDA5$_{iso}$	0.280	**0.079**	0.708	**0.597**	0.294	**0.077**
FDA5$_{dec}$	0.645	**0.242**	**3.006**	4.901	0.583	**0.119**
DIMP2	**11.630**	19.361	**11.379**	19.621	**12.466**	22.124
DMOP2	1.531	**0.798**	**35.126**	40.256	1.165	**0.216**
DMOP2$_{iso}$	0.002	0.002	**0.082**	0.111	0.002	0.002
DMOP2$_{dec}$	**1.044**	2.533	**5.182**	61.288	1.028	**0.324**
DMOP3	0.751	**0.472**	**33.648**	37.936	0.656	**0.121**
HE2	0.365	**0.125**	0.187	**0.125**	0.266	**0.123**
HE7	0.309	**0.051**	0.285	**0.042**	0.311	**0.049**
HE9	0.288	**0.238**	0.273	**0.215**	0.263	**0.245**

For each problem with each parameter, Tr-RMMEDA and RMMEDA get initial populations after each change. MIGD is the average of IGDs of initial populations with one run under 20 changes. The better values obtained by the algorithm are highlighted in bold face.

kinds of changes (C4 and C8), cases that transferred solutions are better for C8 are more than those for C4. As a result, transfer learning works better when the environment change is smaller. To sum up, transfer learning works on problems with small changes and with fixed POS.

One reason for this conclusion is that optimal solutions copied from the previous environment will not become so bad in the new environment, when the changes are small. Furthermore, for problems with fixed POS, optimal solutions will always keep optimal. Besides, the existing transfer learning method is not able to transfer good solutions from the source problem to make them still good in the target problem.

B. Avoiding Transfer on Problems with Fixed POS And Small Environmental Changes

Generally, it is unlikely that the error of transferring good solutions from a problem to anther problem would be zero. Even if the existing Tr-DMOEA is improved, it is unlikely that the error would become zero. As the copied solutions from the previous environment are very good solutions for situations with small changes or problems with fixed POS, it would be very difficult for transferred solutions to beat the copied ones. Therefore, the most intuitive idea to prevent negative results obtained by transfer learning in such cases would be to prevent using transferred solutions.

It has been computationally shown that transfer learning fails on problems with small changes and fixed POS in DMO. In order to improve the performance of transfer learning in DMO, the most intuitive idea is to avoid transfer learning when it fails. In addition, when transfer learning works well, it should be definitely used. However, it is impossible for algorithms to foresee which problem has fixed POS and when the environmental changes are small. To overcome this problem, after each change, transferred solutions and copied ones from the previous environment are firstly combined together. After that, nondominated sort and crowding distance in NSGA-II [12] are used to rank the combined solutions on the new environment. Lastly, solutions with the population size are selected from the combined population as the initial population.

C. Experimental Studies of the Proposed Strategy

This section analyzes the effectiveness of the strategy proposed in section II-B through experiments. Firstly, all experiment setups are the same as those in Section II-A. The only difference is that we compare the proposed strategy against the original transfer learning approach, to check whether it can improve the MIGD values of the initial populations after the changes.

TABLE III
MIGD OF COMBINED SOLUTIONS AND TRANSFERRED SOLUTIONS.

Prob.	C4		C6		C8	
Methods	Tr.	Comb.	Tr.	Comb.	Tr.	Comb.
FDA4	0.082	**0.070**	0.098	**0.057**	0.081	**0.070**
FDA5	0.389	**0.185**	1.294	**0.226**	0.325	**0.181**
FDA5$_{iso}$	0.280	**0.265**	0.708	**0.226**	0.294	**0.260**
FDA5$_{dec}$	0.645	**0.270**	3.006	**0.295**	0.583	**0.353**
DIMP2	11.630	**3.514**	11.379	**3.793**	12.466	**3.807**
DMOP2	1.531	**0.004**	35.126	**18.035**	1.165	**0.004**
DMOP2$_{iso}$	**0.002**	0.004	**0.082**	0.112	**0.002**	0.004
DMOP2$_{dec}$	1.044	**0.019**	5.182	**0.191**	1.028	**0.017**
DMOP3	0.751	**0.003**	33.648	**18.022**	0.656	**0.003**
HE2	0.365	**0.069**	0.187	**0.056**	0.266	**0.061**
HE7	0.309	**0.040**	0.285	**0.036**	0.311	**0.040**
HE9	0.288	**0.226**	0.273	**0.199**	0.263	**0.237**

For each problem with each parameter, combined solutions are the initial population introduced in Section II-B, while transferred solutions are the initial population obtained by Tr-RMMEDA in each change. MIGD is the average of IGDs of these initial populations with one run under 20 changes. The better values that the solution set have are highlighted in bold face.

The specific comparison results of transferred to combined solutions are shown in Table III. The better value that the solution set has is highlighted in bold face. Combined solutions are termed as $Comb$. It is clear that combined solutions are better than transferred ones ($Tr.$) on almost all test problems under different environments. In addition, the Wilcoxon rank sum test with the significance level 0.05 is carried out to indicate significance between results obtained by transferred and combined solutions. MIGD value of $Tr.$ and $Comb.$ for each problem with one parameter is regarded as one observation for the test [13]. The result with h = 1 and p = 1.2532e - 4 shows that the combined solutions are significantly better than transferred solutions in the first generation after change.

III. HOW DOES TRANSFER LEARNING WORK IN DMO?

A. Mathematical Proof that Gaussian Kernels Are Not Ideal

In this section, we briefly introduce the foundations of Tr-DMOEA first. Then the weakness of using the Gaussian kernel will be highlighted. The detailed process of Tr-DMOEA can be found in [5].

In Domain Adaption Learning (DAL) [14], a transfer learning method, it is assumed that a transformation should be found to a latent space where the difference between the distributions of source and target domain is minimized. Once this transformation is found, it can act as a bridge to connect the source domain and the target domain. Then solutions that have been optimized in one domain can be transferred to be good solutions in another domain through this bridge.

The distance between the distributions of the source and target domain can be calculated through the Maximum Mean

Discrepancy (MMD) [15], which evaluates the distance between two distributions in the Reproducing Kernel Hilbert Space. Let p and q be two Borel probability distributions defined on a domain $\mathcal{X}$. $FS = \{Fs_1, ..., Fs_m\}$ and $FT = \{Ft_1, ..., Ft_n\}$ are observations drawn from p and q. Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$. f can be written as $f(x) = \langle \phi(x), f \rangle$ in a RKHS, where $\phi(x) : \mathcal{X} \to \mathcal{H}$. The estimated MMD in RKHS can be calculated as:

$$MMD(\mathcal{F}, p, q) := \left\| \frac{1}{m} \sum_{i=1}^{m} \phi(Fs_i) - \frac{1}{n} \sum_{i=1}^{n} \phi(Ft_i) \right\|_{\mathcal{H}}^2 \quad (1)$$

In Tr-DMOEA, the distribution of the source and target domains in consideration is the distribution of the objective vectors of source and target solutions. Therefore, in Tr-DMOEA, FS and FT are the objective vectors of randomly generated solutions in the source environment s (i.e., the problem before a change) and target environment t (i.e., the problem after a change), respectively. The function ϕ is defined as $\phi(F) = W^T \kappa(F)$, where W is a transformation matrix which maps the objective vector into the latent space, and $\kappa(F)$ is defined as follows, where $\kappa(\cdot, \cdot)$ is a kernel function [16]:

$$\kappa(F) = [\kappa(Fs_1, F), ..., \kappa(Fs_m, F), \kappa(Ft_1, F), ..., \kappa(Ft_n, F)]^T$$

Once W is found, Tr-DMOEA will initialize the population in the target environment with solutions whose objective vectors are close to that of any good solution from the source environment in the latent space. For that, it needs to find solutions t_k whose objective vector is close to that of a solution s_l from POF of the problem in the source environment (POF_s), i.e., whose $||\phi(Fs_l) - \phi(Ft_k)||$ is minimal. We can expand the distance as follows:

$$\begin{aligned} &||\phi(Fs_l) - \phi(Ft_k)|| = ||W^T \kappa(Fs_l) - W^T \kappa(Ft_k)|| \\ =&||W^T[\kappa(Fs_l) - \kappa(Ft_k)]|| = \sum_{j=1}^{d} \Big(\phi_j(Fs_l) - \phi_j(Ft_k)\Big)^2 \\ =&\sum_{j=1}^{d} \left\{ \sum_{i=1}^{m+n} W_{ji} \times \Big(\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k)\Big) \right\}^2 \end{aligned} \quad (2)$$

in which d is the dimension of the latent space; F = Fs when $i \in [1, m]$ and F = Ft when $i \in [m+1, m+n]$.

From the above we can see that the more similar $\kappa(F_i, Fs_l)$ and $\kappa(F_i, Ft_k)$ are, the smaller the distance in the latent space. In the existing Tr-DMOEA, the used kernel function is the Gaussian kernel:

$$\kappa(F, Ft_k) = e^{-(F - Ft_k)^T (F - Ft_k)} \quad (3)$$

Here, we consider bi-objective problems: $Fs_l = (Fs_l^1, Fs_l^2)$; $Ft_k = (Ft_k^1, Ft_k^2)$; $F = (f^1, f^2)$; therefore:

$$\begin{aligned} &\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k) \\ =&e^{-(Fs_l^1 - F_i^1)^2 - (Fs_l^2 - F_i^2)^2} - e^{-(Ft_k^1 - F_i^1)^2 - (Ft_k^2 - F_i^2)^2} \end{aligned} \quad (4)$$

Then, the difference between $\kappa(F_i, Fs_l)$ and $\kappa(F_i, Ft_k)$ is the difference between their exponent:

$$\left| \Big((Fs_l^1 - F_i^1)^2 + (Fs_l^2 - F_i^2)^2\Big) - \Big((Ft_k^1 - F_i^1)^2 + (Ft_k^2 - F_i^2)^2\Big) \right| \quad (5)$$

$$= \left| \Big((Fs_l^1 - F_i^1)^2 - (Ft_k^1 - F_i^1)^2\Big) + \Big((Fs_l^2 - F_i^2)^2 - (Ft_k^2 - F_i^2)^2\Big) \right| \quad (6)$$

To make eq. (5) minimal, both terms of eq (6) should be minimal. To simplify the analysis, only the first term is considered here. When the first term is equal to 0, it can be re-written as follows:

$$\begin{aligned} &\Big|Fs_l^1 - F_i^1\Big| = \Big|Ft_k^1 - F_i^1\Big| \\ \Rightarrow Fs_l^1 - F_i^1 =& Ft_k^1 - F_i^1 \quad \textbf{OR} \quad Fs_l^1 - F_i^1 = F_i^1 - Ft_k^1 \end{aligned} \quad (7)$$

Therefore, solutions for Ft_k^1 are (Ft_k^2 is the similar as Ft_k^1):

$$Ft_k^1 = Fs_l^1 \qquad \textbf{OR} \qquad Ft_k^1 = 2F_i^1 - Fs_l^1 \quad (8)$$

Therefore, objective values of found solutions are close to either those of the solution Fs_l from POF_s or twice over those of randomly generated solutions (F_i). The original intention of Tr-DMOEA was to find a solution in the target environment whose objective vector Ft_k^1 is similar to that of a POF_s solution in the latent space (i.e., the first solution in Eq. (8)). However, there is no reason to believe that a solution in the target domain whose objective vector is similar to a random solution from the source domain in the latent space (i.e., the second solution in Eq. (8)) is going to be a good solution.

In this case, the Gaussian kernel function is not the ideal in present version of Tr-DMOEA.

B. Replacing Gaussian Kernel with Linear Kernel

After reviewing present common kernel functions, we find that the linear kernel functions [17] [18] [19] overcome the problem presented by the Gaussian kernel function explained in section III-A. In the following, the details why the linear kernel overcomes this problem are explained.

Here, the simplest linear kernel is used:

$$\kappa(F, Ft_k) = F^T Ft_k \quad (9)$$

Therefore,

$$\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k) = F_i^1(Fs_l^1 - Ft_k^1) + F_i^2(Fs_l^2 - Ft_k^2) \quad (10)$$

It can be seen from eq (2) when the distance is minimal, $\kappa(F_i, Fs_l) - \kappa(F_i, Ft_k)$ would be close to 0. In this case, $Fs_l^1 = Ft_k^1$ and $Fs_l^2 = Ft_k^2$. Therefore, the searched solution $Ft_k = (Ft_k^1, Ft_k^2)$ will be close to the solution $Fs_l = (Fs_l^1, Fs_l^2)$ in last POF_s, being a potentially good solution to initialize the population in the target domain.

C. Experimental Evaluation of Different Kernels

In section III-B, it has been mathematically proved that the linear kernel function overcomes the problem of the Gaussian kernel. In order to verify the effectiveness of it from the perspective of experiment, specific experiment will be conducted.

In this experiment, two algorithms will be compared. One is Tr-DMOEA with the Gaussian kernel function. Another is Tr-DMOEA with the linear kernel function. For each algorithm, all the experiment setups are the same as those in Section II-A, including used benchmark problems, algorithm parameter setting, dynamics for changes, performance metric and so on. The comparison results of the Gaussian and linear kernel based Tr-DMOEA are shown in Table IV. In the Table, $Gauss.$ and $Lin.$ represent the Gaussian and linear kernel based Tr-DMOEA, respectively.

TABLE IV
MIGD VALUES FOR GAUSSIAN KERNEL-BASED AND LINEAR KERNEL-BASED TR-RMMEDA.

Prob.	C4		C6		C8	
Methods	Lin.	Gauss.	Lin.	Gauss.	Lin.	Gauss.
FDA4	**0.052**	0.082	**0.048**	0.098	**0.052**	0.081
FDA5	**0.238**	0.390	**0.825**	1.295	**0.163**	0.325
FDA5$_{iso}$	**0.204**	0.280	**0.628**	0.708	**0.185**	0.294
FDA5$_{dec}$	**0.464**	0.645	**0.641**	3.006	**0.265**	0.583
DIMP2	**10.909**	11.630	**10.527**	11.379	**12.076**	12.466
DMOP2	**0.006**	1.5313	**33.085**	35.126	**0.003**	1.166
DMOP2$_{iso}$	0.002	0.002	0.087	**0.082**	0.002	0.002
DMOP2$_{dec}$	**0.053**	1.044	9.107	**5.182**	**0.070**	1.028
DMOP3	**0.003**	0.751	35.076	**33.648**	**0.003**	0.656
HE2	**0.115**	0.365	**0.096**	0.187	**0.071**	0.266
HE7	**0.301**	0.308	**0.265**	0.285	**0.300**	0.311
HE9	0.303	**0.288**	0.297	**0.273**	0.294	**0.263**

For each problem with each parameter, Gaussian kernel-based and linear kernel-based Tr-RMMEDA get initial populations after each change. MIGD is the average of IGDs of these initial populations with one run under 20 changes. The better values that the method gets are highlighted in bold face.

It can be seen from the table that when change is small (C8) and medium (C4), Tr-DMOEA with linear kernel function is better than that with Gaussian kernel function except HE9 problem. Plus, when changes are large (C6), there are only 4 out of 12 cases that linear kernel-based Tr-DMOEA is worse than Gaussian kernel-based Tr-DMOEA. As a whole, the linear kernel greatly improves the performance of solution quality after change in the first generation, compared with the Gaussian kernel one. In addition, the Wilcoxon rank sum test with the significance level 0.05 is carried out to indicate significance between results obtained by transferred and combined solutions. MIGD value of $Lin.$ and $Gauss.$ for each problem with one parameter is regarded as one observation data for the test. The result with $h = 1$ and $p = 0.0132$ shows that transferred solutions with the linear kernel are significantly better than those with the Gaussian one in the first generation after change.

IV. IMPROVED TRANSFER LEARNING IN DMO

It has been mathematically and experimentally shown that when and how to transfer is vitally important in DOM. Transfer learning fails on problems with stationary POS and when changes are small. The kernel function also matters in Tr-DMOEA, which contributes the idea of replacing the Gaussian kernel with a linear one. Therefore, in order to make improvements on present Tr-DMOEA, this section proposes novel method for Tr-DMOEA.

A. Proposed Method to Generate An Initial Population

Algorithm 2: Responding strategy based on improved transfer learning. (The highlighted text corresponds to the differences between the original and improved Tr-DMOEA.)

Input: Two DMOPs $F_s(\cdot)$ and $F_t(\cdot)$ in the source and target environments; POS$_s$ and POF$_s$ of the DMOP in the source environment; linear kernel function κ

Output: The responding initial population $\mathbf{P}_{ini}$.

1 Initialization;
2 Randomly sample two solution sets in the search space of problems $F_s(\cdot)$ and $F_t(\cdot)$, as X_s and Y_t;
3 Evaluate X_s and Y_t on their own objective functions to get their objective vectors $\mathbf{F}_s$ and $\mathbf{F}_t$;
4 Use $\mathbf{F}_s$ and $\mathbf{F}_t$ as well as *the linear kernel function* to get the transformation matrix W [5] ;
5 Determine target domain solutions X_{Tr} whose fitness is similar to that of the POF$_s$ solutions in the latent space. *the linear kernel function* is used here to map solutions to the latent space;
6 Calculate the objective values of X_{Tr} and POS$_s$ on problem $F_t(\cdot)$;
7 Sort on the combined populations X_{Tr} and POS$_s$ through nondominated sort and crowding distance;
8 Select the top N solutions from the combined population as $\mathbf{P}_{ini}$, where N is the population size;
9 **Return** $\mathbf{P}_{ini}$.

The main idea in a DMOEA is to generate an initial population such that the population can quickly reach the new optimum after a change. Therefore, in Tr-DMOEA, an initial population is produced through transferring good solutions from the previous environment. In this section, we present a

TABLE V
MIGD VALUES FOR THE ORIGINAL AND IMPROVED TR-RMMEDA

Prob.	C4		C6		C8	
Methods	Tr.	ImTr.	Tr.	ImTr.	Tr.	ImTr.
FDA4	0.082	**0.070**	0.098	**0.058**	0.081	**0.071**
FDA5	0.389	**0.285**	1.294	**0.822**	0.325	**0.249**
FDA5$_{iso}$	**0.280**	0.344	0.708	**0.633**	**0.294**	0.336
FDA5$_{dec}$	0.645	**0.327**	3.006	**0.507**	0.583	**0.254**
DIMP2	**11.630**	12.093	**11.379**	**12.163**	12.466	11.860
DMOP2	1.531	**0.006**	35.126	**32.984**	1.165	**0.003**
DMOP2$_{iso}$	**0.002**	**0.004**	0.082	0.102	**0.002**	0.004
DMOP2$_{dec}$	1.044	**0.052**	**5.182**	10.026	1.028	**0.024**
DMOP3	0.751	**0.004**	33.648	**33.219**	0.656	**0.004**
HE2	0.365	**0.115**	0.187	**0.136**	0.266	**0.112**
HE7	0.309	0.309	**0.285**	0.292	**0.311**	0.319
HE9	0.288	**0.255**	**0.273**	0.275	0.263	**0.254**

For each problem with each parameter, the original and improved Tr-RMMEDA get initial populations after each change. MIGD is the average of IGDs of these initial populations with one run under 20 changes. The better values that the method gets are highlighted in bold face.

TABLE VI
MIGD VALUES OF FOUR DMOEAS WITH THE OPTIMIZATION ALGORITHM RMMEDA ON ALL BENCHMARK PROBLEMS.

Prob.	C1				C2				C3				C4			
Methods	RND	COPY	Trans.	ImTr.	RND	COPY	Tran.	ImTr.	RND	COPY	Trans.	ImTr.	RND	COPY	Trans.	ImTr.
FDA4	0.434	0.105	**0.052**	0.059	0.300	0.080	**0.052**	0.062	0.103	0.065	**0.056**	0.065	0.068	0.060	**0.057**	0.070
FDA5	0.905	0.263	0.705	**0.191**	0.725	**0.164**	0.567	0.188	0.369	**0.099**	0.273	0.175	0.164	**0.087**	0.137	0.165
$FDA5_{iso}$	0.315	**0.066**	0.267	0.227	0.177	**0.066**	0.170	0.192	0.093	**0.068**	0.080	0.276	0.066	0.066	0.066	0.263
$FDA5_{dec}$	2.010	0.968	1.186	**0.313**	1.656	0.314	0.865	**0.267**	0.963	**0.201**	0.780	0.234	0.649	**0.122**	0.494	0.232
DIMP2	9.063	16.351	**8.471**	8.746	7.824	12.310	**6.968**	7.269	5.153	7.916	5.097	**5.087**	4.115	5.381	**3.980**	4.129
DMOP2	1.961	20.295	0.465	**0.003**	0.956	5.994	0.220	**0.003**	0.091	0.143	0.040	**0.003**	0.004	0.009	0.003	**0.003**
$DMOP2_{iso}$	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004
$DMOP2_{dec}$	2.266	2.148	0.538	**0.079**	1.084	2.390	0.420	**0.045**	0.374	0.154	0.147	**0.051**	0.105	0.069	0.061	**0.038**
DMOP3	1.453	13.041	0.243	**0.003**	0.603	3.677	0.095	**0.003**	0.056	0.141	0.015	**0.003**	0.003	0.008	0.004	**0.003**
HE2	2.779	0.451	0.527	**0.140**	2.397	0.268	0.476	**0.110**	1.581	0.116	0.293	**0.068**	0.885	0.085	0.146	**0.063**
HE7	0.172	0.049	0.166	**0.045**	0.129	0.048	0.125	**0.041**	0.076	0.050	0.077	**0.038**	0.053	0.051	0.052	**0.039**
HE9	0.342	0.244	0.294	**0.241**	0.313	0.243	0.273	**0.237**	0.283	0.250	0.254	**0.233**	0.264	0.239	0.244	**0.229**
Prob.	C5				C6				C7				C8			
Methods	RND	COPY	Trans.	ImTr.	RND	COPY	Tran.	ImTr.	RND	COPY	Trans.	ImTr.	RND	COPY	Trans.	ImTr.
FDA4	0.547	0.547	**0.051**	0.053	0.069	0.068	**0.051**	0.057	0.277	0.069	**0.052**	0.063	0.066	0.058	**0.057**	0.071
FDA5	1.061	2.166	0.558	**0.345**	0.271	0.457	0.195	**0.179**	0.675	**0.123**	0.386	0.183	0.183	**0.084**	0.114	0.164
$FDA5_{iso}$	0.243	0.212	**0.176**	0.186	0.061	0.061	0.062	0.180	0.171	**0.066**	0.164	0.192	0.065	0.066	**0.065**	0.258
$FDA5_{dec}$	1.127	2.049	0.948	**0.426**	0.328	0.472	0.270	**0.206**	1.566	0.219	0.945	**0.127**	0.609	**0.098**	0.537	0.144
DIMP2	7.845	13.596	7.338	**7.230**	4.341	7.575	**4.202**	4.289	8.176	15.904	**7.218**	7.281	4.126	7.204	4.097	**3.929**
DMOP2	26.944	37.821	23.744	**18.039**	18.308	28.273	18.899	**18.008**	0.874	1.528	0.295	**0.003**	0.004	0.005	0.003	**0.003**
$DMOP2_{iso}$	0.110	0.110	0.110	**0.090**	0.110	0.110	0.110	0.112	0.002	0.002	0.002	0.004	0.002	0.002	0.002	0.004
$DMOP2_{dec}$	2.742	59.767	1.116	**0.797**	0.232	36.583	0.238	**0.224**	1.325	0.300	0.402	**0.015**	0.143	0.020	0.050	**0.012**
DMOP3	25.957	35.728	22.625	**18.016**	18.280	27.185	18.654	**18.011**	0.459	0.269	0.120	**0.003**	0.003	0.003	0.003	0.003
HE2	1.807	0.258	0.281	**0.071**	0.726	0.084	0.085	**0.061**	2.342	0.279	0.432	**0.094**	0.907	0.085	0.123	**0.061**
HE7	0.116	0.042	0.114	**0.035**	0.047	0.041	0.048	**0.035**	0.131	0.052	0.130	**0.042**	0.053	0.046	0.051	**0.040**
HE9	0.287	0.214	0.249	**0.203**	0.234	0.221	0.219	**0.201**	0.316	0.247	0.275	**0.234**	0.264	0.248	0.243	**0.228**

For each problem with each parameter, all algorithms get optimized populations by RMMEDA in the last generation of each change. MIGD is the average of IGDs of these populations with one run under 20 changes. The best values that the algorithm obtains are highlighted in bold face.

method that makes use of the strategies proposed in sections II-B and III-B to improve Tr-DMOEA's solutions in the initial population after the changes. Firstly, in the process of transfer learning, the Gaussian kernel will be replaced with a linear one, as described in section II-B. Secondly, considering that it is difficult to judge when changes are small and which problem has fixed POS for a DMOEA, transferred solutions and solutions copied from the previous environment are combined together, as described in section III-B. The initial population is selected from the combined populations and then regarded as the initial population for the problem in next environment. The detailed procedures of improved Tr-DMOEA are shown in Algorithm 1. The differences with regarding to the original Tr-DMOEA are highlighted in the pseudocode. This algorithm can be embedded into any population based evolutionary algorithms.

B. Evaluation through Computational Studies

In order to evaluate the effectiveness of the proposed improved Tr-DMOEA, computational studies are conducted in this section. Firstly, we compare the performance of initial population produced by the original transfer learning and improved one in DMO following a similar procedure to that used in Sections II-C and III-C. Then, the performance of populations obtained by several compared methods after optimization is stated. The reason why we are now also comparing the results after optimization is to verify whether the improved Tr-DMOEA helps to solve DMOPs, compared with other state-of-the-arts.

1) Solutions Quality Comparison in the First Generation: Similar to previous experiments, comparison results of solutions after the change produced by the original and improved Tr-RMMEDA are shown in Table V. Experimental setup is the same as those in Section II-A. It can be observed from the table that results of the improved Tr-RMMEDA are better than the original one on most of investigated problems. In addition, the Wilcoxon rank sum test with the significance level 0.05 is carried out to indicate significance between results obtained by transferred and combined solutions. MIGD value of $Tr.$ and $ImTr.$ for each problem with one parameter is regarded as one observation data for the test. The result with h = 1 and p = 0.0269 shows that the transferred solutions through the original Tr-DMOEA are significantly better than those through the improved Tr-DMOEA in the first generation after change.

2) Solutions Quality Comparison after Optimization: Here, the experiment is conducted to verify the effect of the proposed method at the final generation after the changes. The CEC 2015 Benchmark problems [8] are selected as the test problems. For the parameters regarding the dynamics of these problems, **all parameters** shown in Table I are used. It aims to show the performance of all compared methods under different dynamics. There are 20 environmental changes. The population is force to run 50 generations at the beginning of the algorithm, which enables the population to converge.

Experimental setup is as follows:

1) Population size N is set as 200, as in previous work [5];
2) Three investigated population-based algorithms, the fast and elitist multi-objective genetic algorithm (NSGA-II) [12], multi-objective particle swarm optimization algorithm (MOPSO) [20] and regularity model-based multi-objective estimation of distribution algorithm (RMMEDA) [9], are regarded as the optimization algorithms. The DMOEAs with the original transfer learning are termed as Tr-DMOEAs. Similarly, DMOEAs with proposed method are written as ImTr-DMOEAs.
3) Compared algorithms: For each ImTr-DMOEA, there are three compared DMOEAs: the original Tr-DMOEA, the static MOEA (COPY-DMOEA) and one with ran-

dom response strategy (RND-DMOEA). The reason for using three different optimization algorithms is to test the performance sensitivity of transfer learning based method in three different type of algorithms.

4) Parameters of different algorithms are set as the same in their original papers [5] [7] [12] [20].

Inverted Generation Distance (IGD) is used to assess the performance of two solutions sets, which can measure the convergence and diversity of a solution set, while from different aspects. At the last generation of each change, every algorithm will get a Pareto optimal front on each problem. The POF is used to calculate the IGD values. MIGD is the average of IGD values under 20 changes. Therefore, MIGD results of four compared DMOEAs with RMMEDA and NSGA-II on all test problems are shown in Tables VI-VII. Due to space limitations, the table showing MIGD results of four compared DMOEAs with MOPSO is omitted here and put in a supplementary material, which can be found in http://www.escience.cn/people/gruan/index.html. The results obtained by four DMOEAs with MOPSO are similar to those with RMMEDA.

Additionally, in order to show the significant superiority of the proposed method to other algorithms, Friedman and Nemenyi statistical tests are conducted on all benchmark problems regarding MIGD of 12 algorithms (4 responding strategies with 3 EAs). The MIGD value obtained by a given algorithm on one problem with one parameter setting is regarded as an observation to compose that algorithm's group for the test, following Demsar's guidelines [13]. Therefore, there are 96 (12 problems and 8 parameters) observations in each group. Friedman detects significant differences in average accuracy with a p-value of 1.3726e-55. The Nemenyi post-tests are shown in Figure 1, and are discussed over the next sub-sections. According to the test, average accuracy of the ImTr-RMMEDA is significantly better than that of the other approaches except Tr-RMMEDA and ImTr-MOPSO.

a) Impact of EAs on Different DMOEAs: Some DMOEAs with different EAs have different performance, while others have the similar performance, which can be seen from Figure. 1. As a whole, MIGD values that RMMEDA obtains are better than those from MOPSO in most cases regarding all problems and different dynamics, while NSGA-II gets the worst MIGD results. For example, for problem FDA4 with the parameter setting C2, each DMOEA with RMMEDA has better MIGD than that of NSGA-II, no matter what the MIGD order of different DMOEAs with the same EA. This shows that RMMEDA are more suitable to solve these DMOPs than other two EAs. The above observations are confirmed by the Friedman and Nemenyi tests in Figure 1.

When transferred solutions are not converged, solutions with Gaussian kernel-based transfer have better diversity, enabling better results to be achieved after the change than when using linear kernel. That is why the improved Tr-NSGA-II is worse than the original Tr-NSGA-II.

b) Performance of DMOEAs on Different Benchmarks: In general, it is clear from Tables VI-VII that transfer learning based DMOEAs are all better than static MOEAs and DMOEAs with random initial population. This shows that transferred solutions can maintain the balance of convergence and diversity to some degree, compared with static MOEAs and random DMOEAs. When comparing the performance of Tr-DMOEAs and ImTr-DMOEAs, ImTr-DMOEAs are basically better than Tr-DMOEAs. Specifically, for HEs problems, ImTr-DMOEAs are all better than Tr-DMOEAs except HE2 and HE7 when the EA is NSGA-II. This shows that transfer learning based DMOEA with the linear kernel and combined solutions is more capable of solving DMOPs with fixed POS.

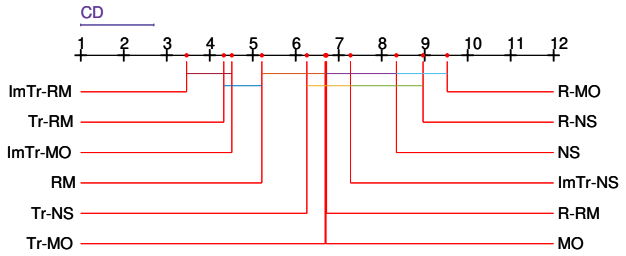

Fig. 1. Friedman ranking among all DMOEAs from left to right, 4 responding strategies with respect to 3 EAs including RMMEDA, NSGA-II and MOPSO (referred as RM, NS and MO for simplicity). Any pair of approaches whose distance between them is larger than CD are considered to be different.

Regarding tri-objective optimization problems, ImTr-DMOEAs performs better than Tr-DMOEAs on FDA5$_{iso}$ and FDA5$_{dec}$ while worse than Tr-DMOEAs on FDA4 and FDA5. This implies that the improved Tr-DMOEAs can solve tri-objective problems with complex property such as isolated or deceptive POF. For DIMP2 whose decision variable has its own rate of change, it is difficult for any algorithm to solve, compared with other problems. The original Tr-DMOEA shows its superiority on this problem. In terms of other bi-objective problems, the Tr-DMOEAs with linear kernel function are all superior to those with the Gaussian kernel. This demonstrates that the improved Tr-DMOEA is better capable of solving tri-objective problems except the one with very complicated features like DIMP2, for which all algorithms struggle.

c) Influence of Dynamics on Tr-DMOEAs: Overall, it is clear from Tables VI-VII that MIGD values obtained by all DMOEAs become better with the increase of changing frequency, when they are under same changing severity. The reason is obvious, which is that the more the iterations within each change, the better the performance. Additionally, no matter what kind of changing sizes are, two kinds of Tr-DMOEAs are all better than the random and static ones. The only difference between the original and the improved Tr-DMOEAs is their performance with different EAs, which has been introduced in section IV-B2a. Therefore, it can be concluded that the improved Tr-DMOEA can address most investigated problems no matter what kinds of dynamics are, as long as solutions before the change have been converged.

V. Conclusion

This paper studies transfer learning in DMO, analyzing when and how transfer learning works in DMO. It has been computationally observed that transfer learning works poorly on problems with fixed POS and when environmental changes

TABLE VII
MIGD VALUES OF FOUR DMOEAS WITH THE OPTIMIZATION ALGORITHM NSGA-II ON ALL BENCHMARK PROBLEMS.

Prob.	C1				C2				C3				C4			
Methods	RND	COPY	Trans.	ImTr.	RND	COPY	Tran.	ImTr.	RND	COPY	Trans.	ImTr.	RND	COPY	Trans.	ImTr.
FDA4	0.676	2.329	**0.314**	0.365	0.657	2.376	**0.237**	0.391	0.429	2.134	**0.152**	0.293	0.280	0.777	**0.118**	0.263
FDA5	1.039	1.888	**0.362**	0.812	0.947	2.147	**0.318**	0.733	0.693	1.812	**0.321**	0.519	0.419	1.106	**0.291**	0.327
FDA5$_{iso}$	0.279	**0.197**	0.790	0.290	0.176	**0.133**	0.416	0.141	0.099	0.113	0.229	**0.095**	**0.098**	0.100	0.135	**0.098**
FDA5$_{dec}$	1.671	2.536	**0.501**	1.075	1.428	2.415	**0.409**	0.966	0.877	1.250	**0.380**	0.696	0.722	0.636	**0.300**	0.409
DIMP2	9.683	12.735	**5.990**	9.502	7.685	10.792	**4.233**	7.111	4.780	8.172	**3.312**	4.394	3.976	5.292	**2.404**	3.724
DMOP2	2.492	29.246	**0.744**	1.618	2.214	24.647	**0.560**	1.730	0.476	5.311	**0.147**	0.572	0.164	0.099	**0.047**	0.130
DMOP2$_{iso}$	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003
DMOP2$_{dec}$	2.806	5.644	**0.825**	1.043	2.071	5.083	**0.640**	1.263	0.858	1.053	**0.367**	0.664	0.486	0.615	**0.244**	0.384
DMOP3	2.267	24.577	**0.619**	1.102	1.769	26.646	**0.338**	1.030	0.463	3.987	**0.128**	0.510	0.083	0.153	**0.056**	0.100
HE2	2.821	1.834	**0.312**	2.409	2.557	1.296	**0.264**	1.982	1.155	0.784	**0.211**	1.247	0.464	0.367	**0.172**	0.401
HE7	0.228	0.086	**0.073**	0.163	0.194	0.075	**0.074**	0.104	0.138	**0.088**	0.096	0.093	0.090	0.072	0.077	**0.070**
HE9	0.363	**0.296**	0.398	0.344	0.351	**0.298**	0.371	0.333	0.328	**0.295**	0.324	0.316	0.303	**0.287**	0.302	0.311
Prob.	C5				C6				C7				C8			
Methods	RND	COPY	Trans.	ImTr.	RND	COPY	Tran.	ImTr.	RND	COPY	Trans.	ImTr.	RND	COPY	Trans.	ImTr.
FDA4	1.027	0.559	**0.243**	0.894	0.617	0.593	**0.101**	0.478	0.644	1.966	**0.258**	0.362	0.283	0.592	**0.102**	0.227
FDA5	1.185	0.542	**0.419**	0.711	0.709	0.490	**0.337**	0.368	0.896	1.176	**0.227**	0.806	0.325	0.798	**0.243**	0.421
FDA5$_{iso}$	0.258	**0.173**	0.251	0.284	**0.125**	0.140	0.173	0.166	0.205	0.173	0.217	**0.138**	**0.096**	0.100	0.248	**0.096**
FDA5$_{dec}$	1.399	1.730	1.021	**0.485**	1.181	0.921	0.774	**0.369**	1.596	2.046	**0.390**	0.575	0.426	0.773	0.323	**0.272**
DIMP2	7.634	6.109	**4.049**	7.088	3.737	4.046	**2.297**	4.075	8.105	9.614	**4.430**	7.385	4.241	6.098	**2.645**	3.254
DMOP2	6.296	**3.114**	3.465	7.669	0.702	**0.237**	0.507	0.426	2.168	21.833	**0.389**	1.312	0.108	0.078	**0.043**	0.099
DMOP2$_{iso}$	0.119	0.122	0.137	**0.097**	0.119	**0.117**	0.140	0.129	0.003	0.003	0.003	0.003	0.003	0.003	0.003	0.003
DMOP2$_{dec}$	8.249	10.087	**2.993**	8.118	1.368	13.831	**0.898**	1.463	1.970	1.172	**0.459**	1.205	0.246	0.280	**0.178**	0.310
DMOP3	10.584	**0.293**	1.978	5.253	0.308	**0.298**	3.350	3.979	1.360	25.525	**0.270**	1.087	0.127	0.156	**0.029**	0.064
HE2	1.735	0.340	**0.094**	0.362	1.087	0.249	**0.073**	0.184	2.655	0.461	**0.247**	1.572	0.926	0.295	**0.172**	0.355
HE7	0.174	0.075	**0.068**	0.090	0.079	0.051	0.042	**0.041**	0.197	**0.079**	0.099	0.128	0.101	**0.055**	0.056	0.065
HE9	0.319	**0.243**	0.296	0.280	0.271	**0.242**	0.259	0.251	0.350	**0.270**	0.361	0.341	0.302	0.293	0.308	**0.289**

For each problem with each parameter, all algorithms get optimized populations by NSGA-II in the last generation of each change. MIGD is the average of IGDs of these populations with one run under 20 changes. The best values that the algorithm obtains are highlighted in bold face.

are small. It has also been shown that the Gaussian kernel function in the existing method Tr-DMOEA [5] is inadequate for DMO. Based on these two observations, a new method has been proposed regarding avoiding transfer learning when it fails and replacing the Gaussian kernel with a linear one. Experimental results have shown that our proposed method is effective in solving DMOPs, compared with other state-of-the-art algorithms.

In the future, a potential work is to find another kernel function that can achieve non-linear transfers and does not have the weakness of the Gaussian kernel. In addition, other transfer learning methods in the field of machine learning can also be studied to solve DMOPs. At last, it is important to explore real-world applications of transfer learning based DMO, e.g., in smart manufacturing and smart logistics.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 766186. The work was also supported by the Program for Guangdong Introducing Innovative and Enterpreneurial Teams (Grant No. 2017ZT07X386), Shenzhen Peacock Plan (Grant No. KQTD2016112514355531) and the Program for University Key Laboratory of Guangdong Province (Grant No. 2017KSYS008).

REFERENCES

[1] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2009.

[2] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: toward evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, 2015.

[3] B. Da, A. Gupta, and Y.-S. Ong, "Curbing negative influences online for seamless transfer evolutionary optimization," *IEEE T. Cybern.*, no. 99, pp. 1–14, 2018.

[4] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: test cases, approximations, and applications," *IEEE Trans. Evol. Comput.*, vol. 8, no. 5, pp. 425–442, 2004.

[5] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen, "Transfer learning-based dynamic multiobjective optimization algorithms," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 501–514, 2017.

[6] G. Ruan, G. Yu, J. Zheng, J. Zou, and S. Yang, "The effect of diversity maintenance on prediction in dynamic multi-objective optimization," *Appl. Soft. Comput.*, vol. 58, pp. 631–647, 2017.

[7] A. Zhou, Y. Jin, and Q. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization," *IEEE T. Cybern.*, vol. 44, no. 1, pp. 40–53, 2013.

[8] M. Helbig and A. Engelbrecht, "Benchmark functions for cec 2015 special session and competition on dynamic multi-objective optimization," *Dept. Comput. Sci., Univ. Pretoria, Pretoria, South Africa, Rep*, 2015.

[9] Q. Zhang, A. Zhou, and Y. Jin, "Rm-meda: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 41–63, 2008.

[10] M. R. Sierra and C. A. C. Coello, "Improving pso-based multi-objective optimization using crowding, mutation and-dominance," in *EMO*. Springer, 2005, pp. 505–519.

[11] Q. Li, J. Zou, S. Yang, J. Zheng, and G. Ruan, "A predictive strategy based on special points for evolutionary dynamic multi-objective optimization," *Soft Comput.*, vol. 23, no. 11, pp. 3723–3739, 2019.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002.

[13] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, no. Jan, pp. 1–30, 2006.

[14] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, no. 1-2, pp. 151–175, 2010.

[15] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A hilbert space embedding for distributions," in *International Conference on ALT*. Springer, 2007, pp. 13–31.

[16] J. Shawe-Taylor, N. Cristianini *et al.*, *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

[17] J. Nam, S. J. Pan, and S. Kim, "Transfer defect learning," in *2013 35th ICSE*. IEEE, 2013, pp. 382–391.

[18] B. Chen, W. Lam, I. Tsang, and T.-L. Wong, "Extracting discriminative concepts for domain adaptation in text mining," in *Proceedings of the 15th ACM SIGKDD*. ACM, 2009, pp. 179–188.

[19] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, "Cross-domain sentiment classification via spectral feature alignment," in *Proceedings of the 19th WWW*. ACM, 2010, pp. 751–760.

[20] C. C. Coello and M. S. Lechuga, "Mopso: A proposal for multiple objective particle swarm optimization," in *Proceedings of the 2002 CEC (Cat. No. 02TH8600)*, vol. 2. IEEE, 2002, pp. 1051–1056.