Transfer Learning-Based Dynamic Multiobjective Optimization Algorithms

Min Jiang, *Senior Member, IEEE*, Zhongqiang Huang, Liming Qiu, Wenzhen Huang, and Gary G. Yen, *Fellow, IEEE*

***Abstract*—One of the major distinguishing features of the dynamic multiobjective optimization problems (DMOPs) is that optimization objectives will change over time, thus tracking the varying Pareto-optimal front becomes a challenge. One of the promising solutions is reusing "experiences" to construct a prediction model via statistical machine learning approaches. However, most existing methods neglect the nonindependent and identically distributed nature of data to construct the prediction model. In this paper, we propose an algorithmic framework, called transfer learning-based dynamic multiobjective evolutionary algorithm (EA), which integrates transfer learning and population-based EAs to solve the DMOPs. This approach exploits the transfer learning technique as a tool to generate an effective initial population pool via reusing past experience to speed up the evolutionary process, and at the same time any population-based multiobjective algorithms can benefit from this integration without any extensive modifications. To verify this idea, we incorporate the proposed approach into the development of three well-known EAs, nondominated sorting genetic algorithm II, multiobjective particle swarm optimization, and the regularity model-based multiobjective estimation of distribution algorithm. We employ 12 benchmark functions to test these algorithms as well as compare them with some chosen state-of-the-art designs. The experimental results confirm the effectiveness of the proposed design for DMOPs.**

Manuscript received November 30, 2016; revised April 19, 2017, July 20, 2017, and September 20, 2017; accepted November 3, 2017. Date of publication November 8, 2017; date of current version July 27, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61003014 and Grant 61673328, and in part by the Foundation of Xiamen University's President under Grant 20720150150. The work of M. Jiang was supported in part by the China Scholarship Council under Grant 20150631505, and in part by the Oklahoma State University. *(Corresponding author: Gary G. Yen.)*

M. Jiang is with the Department of Cognitive Science and Technology and the Fujian Key Laboratory of Machine Intelligence and Robotics, Xiamen University, Xiamen 361005, China, and also with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078 USA.

Z. Huang is with the Institute of Innovation Research, Sangfor Technologies, Shenzhen 518071, China.

L. Qiu is with the Department of Cognitive Science and Technology and the Fujian Key Laboratory of Machine Intelligence and Robotics, Xiamen University, Xiamen 361005, China.

W. Huang is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: minjiang@xmu.edu.cn).

G. G. Yen is with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078 USA (e-mail: gyen@okstate.edu).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This material is a PDF file containing 12 tables which record the MIGD values of the six algorithms, NSGA-II, Tr-NSGA-II, MOPSO, Tr-MOPSO, RM-MEDA, and Tr-RM-MEDA, running on the 12 testing functions under eight environments, from C1 to C8. This material is 66.9 KB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2017.2771451

***Index Terms*—Dimensionality reduction, domain adaption, dynamic multiobjective optimization, evolutionary algorithm (EA), transfer learning.**

I. Introduction

ONE OF the essential characteristics of dynamic multiobjective optimization problems (DMOPs) [1] is that objective functions will vary over time or under different environments. This underlying problem characteristic bears significant implications for real-world applications [2]. A good example is dynamic portfolio optimization problem, which is common in deregulated electricity markets in which the operations of different power stations are controlled and coordinated to maximize profit while minimizing risk. There are various uncertainties in a deregulated electricity market, including spot market prices, load obligations, and strip/option prices [3]. The values for some of these factors change over time, and it is ordinary to optimize for the market price every hour. However, the optimization approaches including population-based metaheuristics often find extreme difficulty to address the challenge since that the Pareto-optimal front (POF) of a DMOP may change when the environment changes. Solving the DMOPs efficiently and effectively has become an important research issue in evolutionary computation community [4], [5].

In recent years, a great deal of progress has been made and different types of algorithms have been proposed. In all of these methods, one class of approaches, the prediction-based, has gained much attention. This class of approaches allows evolutionary algorithm (EA) and machine learning to be seamlessly integrated. After deriving a prediction model via machine learning techniques, the EAs can sustain the needed performance even if the environment changes over time. For example, Simões and Costa [6] proposed a memory-based EA which introduced two kinds of prediction models. The first one used the linear/nonlinear regression model to predict when the environment would change while the second model was based on Markov chains which was used to forecast changes. Rossi *et al.* [7] suggested integrating motion information into an EA, such that the algorithm can track a time-changing optimum. Stroud [8] proposed a Kalman-extended genetic algorithm (KGA), and this algorithm was developed to determine when to re-evaluate an existing individual, when to produce a new individual, and which individual to re-evaluate.

The basic idea of these methods is "keeping track of good (partial) solutions in order to reuse them under periodically changing environment" [7]. If we consider this view from

a statistical point of view, this idea implies that the solutions of a dynamic optimization problem obey an identical distribution. In other words, the solutions which are used to construct the prediction model and the solutions forecasted by the prediction model meet the independent identical distribution (IID) hypothesis to some extent. This assumption undoubtedly simplifies the complexity of the problem; however, we have to understand there is an appreciable difference between the good, but out-of-date solutions and the proper and newly generated solutions, especially under a dynamic environment. That is to say, the changing POF may lead to the different distributions of the training samples and the predicted samples, and this problem is very difficult for the traditional machine learning methods.

The findings from machine leaning community [9] already showed that a prediction model built by traditional machine learning methods leaves much room to be desired when the training samples and the predicted samples fail to meet the IID hypothesis. Transfer learning [9] allows the distribution of data used in training and testing to be different and it is becoming a useful weapon to overcome this difficulty. Therefore, the dynamic multiobjective optimization algorithms based on traditional machine learning methods, especially the prediction-based algorithms, can also have significant performance improvements by overcoming the limitation caused by the IID, and transfer learning approach is a powerful tool we can use to improve performance of EAs for DMOPs.

In this paper, we argue that integrating transfer leaning approaches [9] into an EA can offer significant benefits to performance and robustness for designing better dynamic multiobjective EAs (DMOEAs). We adopt a domain adaptation method,[1] called transfer component analysis (TCA) [10], to construct a prediction model. This model uses the gained knowledge of finding Pareto optimal solutions, but not the population, to generate an initial population pool for the optimization function at the next time. Based on this initial population pool, the optima of the changed environment can be found more efficiently and effectively. The proposed domain adaptation learning (DAL) approach can be easily incorporated into any evolutionary-based multiobjective optimization algorithms. Please note that in this paper, the dynamic refers to that objective functions will vary over time or under different environments.

Indeed, how to detect and identify dynamic changes is a crucial part of solving DMOPs. However, in this paper, our focus is placed solely on how EA can quickly reoptimize a given dynamic optimization problem once the change is been detected and identified. This is in a similar spirit as those studies in fault tolerant control where focus is placed exclusively on designing a controller capable of accommodating the dynamic changes, leaving the fault detection and identification to be addressed separately.

The contribution of this paper is the integration between transfer learning and classical evolutionary multiobjective optimization algorithms. This combination provides two benefits. *First*, the advantages of the EAs are preserved in the improved design for DMOPs. *Second*, the proposed design can significantly improve the search efficiency via reusing past experience which is critical for solving the DMOPs. An algorithm requires too much computing resources, often making it difficult to solve large-scale problems. The experiments also validate the assumption that the population plays a very important role for tracking dynamic optima, and Dang *et al.* [11], [12] proved it from the theoretical point of view.

The rest of this paper is organized as follows. In Section II, we will introduce some basic concepts of dynamic optimization problems first and then discuss the existing works in this field. At the beginning of Section III, we will present some background on transfer learning, DAL, and then introduce the TCA method in detail. After that we will propose the transfer learning-based DMOEA (Tr-DMOEA). In Section IV, we will present the experimental results of incorporating our approach to improve three well-known MOEAs: 1) nondominated sorting genetic algorithm II (NSGA-II); 2) multiobjective particle swarm optimization (MOPSO); and 3) regularity model-based multiobjective estimation of distribution algorithm (RM-MEDA), specifically for DMOPs and all of the algorithms were tested on the IEEE CEC 2015 benchmark problems set. In Section V, we will draw a summary of this paper and outline the future research directions.

II. Preliminary Studies and Related Research

A. Dynamic Multiobjective Optimization

Formally, a DMOP is defined as

$$\begin{aligned} \textbf{minimize} \quad & F(x,t) = \langle f_1(x,t), f_2(x,t), \ldots, f_M(x,t)\rangle \\ \text{s.t.} \quad & x \in \Omega \end{aligned}$$

where $x = <x_1, x_2, \ldots, x_n>$ is the decision vector and t is the time or environment variable. $f_i(x,t) : \Omega \to \mathbb{R}$ $(i = 1, \ldots, M)$. $\Omega = [L_1, U_1] \times [L_2, U_2] \times \cdots \times [L_n, U_n]$. $L_i, U_i \in \mathbb{R}$ are the lower and upper bounds of the ith decision variable, respectively. Please note that dynamic environments can be classified in different ways. For an in-depth description, the readers are referred to [6].

Definition 1 (Dynamic Decision Vector Domination): At time t, a decision vector x_1 Pareto dominate another vector x_2, denoted by $x_1 \succ_t x_2$, if and only if

$$\begin{cases} \forall i = 1, \ldots, M, & f_i(x_1,t) \le f_i(x_2,t) \\ \exists i = 1, \ldots, M, & f_i(x_1,t) < f_i(x_2,t). \end{cases} \tag{1}$$

Definition 2 (Dynamic Pareto-Optimal Set): Both x and x^* are decision vectors, and if a decision vector x^* is said to be nondominated at time t if and only if there is no other decision vector x such that $x \succ_t x^*$ at time t. The dynamic Pareto-optimal set (DPOS) is the set of all Pareto optimal solutions at time t, that is

$$\text{DPOS} = \left\{x^* | \nexists x,\ x \succ_t x^*\right\}.$$

Definition 3 (Dynamic POF): At time t, the dynamic POF (DPOF) is the corresponding objective vectors of the DPOS

$$\text{DPOF} = \left\{F\left(x^*, t\right) | x^* \in \text{DPOS}\right\}.$$

For an ideal dynamic multiobjective algorithm, it must be able to find a set of solutions as close as possible to the

[1] A branch of transfer learning.

changing POF and at the same time, the set of solutions should be as diverse as possible.

B. Related Works

Much progress [4], [13]–[15] has been made in the DMOPs field in recent years, and most existing algorithms can be classified into the following categories: increasing/maintaining diversity methods, memory-based methods, multipopulation-based methods, and prediction-based methods.

The increasing diversity methods tend to add variety to the population by using a certain type of methodology when the environment change was detected. For example, Cobb [16] proposed the triggered hypermutation method, and the basic idea of this method is that when change is identified, the mutation rate would be increased immediately, and this would make the converged population divergent again. This approach calls for some improvements, and one of them is that the mutation rate is in a state of uncontrolled change during the whole process, and this ultimately results in reduced performance of the algorithm. Therefore, Vavak *et al.* [17] presented a mutation operator, called variable local search (VLS), to address the problem. The strategy that the VLS adopted was to gradually increase the mutation rate. Woldesenbet and Yen [18] proposed a dynamic EA which relocates the individuals based on their change in function value due to the change in the environment and the average sensitivities of their decision variables to the corresponding change in the objective space. This approach can avoid the drawbacks of previous methods to a certain extent.

Most of the methods in the maintaining diversity category assume that avoiding population convergence can help the algorithm track the changing optimum as soon as possible, and maintain diversity as one of the effective means to that end. Grefenstette [19] proposed a random immigrants genetic algorithm (RIGA), and the method replaces some individuals in the population randomly. The idea of the RIGA is that introducing new genetic materials into the population can avoid the whole population converging toward a small area in the process of evolution. However, the drawback of the primitive immigrant method was the fitness values of the introduced individuals were usually low, so large amounts are eliminated during the selection stage, and as a result, it is very difficult to introduce different genes into the population. For solving this problem, Yang and Tinós [20] and Mavrovouniotis and Yang [21] proposed the hybrid immigrants scheme, memory-based immigrants [22] and elitism-based immigrants [22], and these methods are effective for dealing with periodically changing DMOPs. However, if the knowledge about the dynamic environment is limited, they would obtain a greatly reduced efficiency.

Dynamic NSGA-II (DNSGA-II) proposed by Deb *et al.* [23] also shares a similar idea, and this method handles the DMOPs by introducing diversity when change is detected. There are two versions of the proposed DNSGA-II and they are, respectively, known as DNSGA-II-A and DNSGA-II-B. In the DNSGA-II-A, the population is replaced by some individuals with new randomly created solutions, while in the DNSGA-II-B, diversity was guarded by replacing a percentage of the population with mutated solutions.

Memory mechanism enables EAs to record past information, and when it detects changes have occurred, stored information can be reused to improve the performance of the algorithm. Existing research showed that memory-based approaches tend to be more effective on the DMOPs with periodically changing environments.

Branke [24] proposed a direct memory scheme where the best individuals in the population will be saved in an archive, and when the algorithm detects a change, those saved individuals can be retrieved and returned to the population to replace the same number of individuals. Goh and Tan [25] proposed a co-evolutionary multiobjective algorithm which hybridizes competitive and cooperative mechanisms to solve the DMOPs. In this algorithm, the out-of-date archived solutions are replaced by an external population. Wang and Li [26] presented an algorithm called MS-MOEA to tackle the challenges of DMOPs. In the method, adaptive genetic and differential operators were used to speed up the convergence speed and a Gaussian local search operator was employed to prevent from premature convergence. At the same time the fast hyper-volume strategy [27] was proposed to achieve a better starting population when changes occur frequently. The above methods meet some problems, e.g., slow convergence and poor diversity, when the environment changes. As a result Azzouz *et al.* [28] proposed an adaptive hybrid population management strategy using memory, local search, and random strategies to effectively handle environment dynamicity in DMOPs. The special feature of this algorithm is that it can adjust the number of memory and random solutions to be used according to the change severity.

The multipopulation strategy is considered as one efficient solution for the DMOPs, especially for the multiple peaks and the competing peaks problems. Branke *et al.* [29] proposed the self-organizing scouts method, and this method splits the population into scout and base populations, and the two populations are responsible for exploitation and exploration, respectively. In other words, the base population searches for the optimal solution and if the base population finds a peak, then the scout population is generated to track the change of this new peak. Li and Yang [30] employed a multipopulation particle swarm optimization (PSO) algorithm to solve multiple peaks problems. In their method, a population uses evolutionary programming, which shows a better global search ability when compared to other EAs, to explore the most hopeful areas in the whole search space, and at the same time, several subpopulations use the fast PSO algorithm to find the local optima. Yang and Li [31] used hierarchical clustering technique to divide the population into different subpopulations, and the main advantage of this design is that the initial individuals of the subpopulations can be generated automatically according to the fitness landscape.

In general, a good dynamic optimization algorithm should be able to track the changing optimal solution even under high severity and frequency of changes. It must be able to reuse as much information available from previous generations to speedup the optimization search. As a result, in recent years

the prediction-based DMOPs algorithms have received much attention. This class of methods predicts the state of the changing environment typically using the information that already exists and some forms of machine learning techniques, and then makes a decision such that the algorithms can accommodate the changes in advance. This is one of the reasons why the prediction-based approaches can improve performance of an algorithm handling the DMOPs, compared with other types of approaches.

Bosman [32] believed that the decision made at one point would affect the optima obtained in the future, so for the dynamic optimization problems, he proposed an algorithmical framework which integrated machine learning, statistic learning, and evolutionary computation, and this framework can effectively predict what the state of environment is going to be. Rossi *et al.* [7] suggested that the state of an optimum should contain the location and the speed information, so the Kalman filter technique can be used to estimate the state of the system and the error. The authors proposed an EA to measure the state of the past optimum and then use the Kalman filter to obtain an estimated value of the optimum in the next time instance.

Stroud [8] proposed the KGA, and the basic idea of the KGA was that two types of uncertainties surrounded the estimated value of an individual in a dynamical environment. The first type of uncertainty is produced by the dynamic of the environment while the second type was related to the evaluation of individuals. For the different situations, the KGA has two different ways to update the covariances, and uses the Kalman filter technique to predict the two uncertainties which allows the algorithm to work well in a dynamic environment.

Zhou *et al.* [33] presented an algorithm, called population prediction strategy, to predict a whole population instead of predicting some isolated points. There are two key concepts here: 1) center point and 2) manifold. Whenever a change is detected, the algorithm uses a sequence of center points obtained from the search progress to predict the next center point, and at the same time, the previous manifolds are used to estimate the next manifold. The main problem of this method is that, it is difficult to obtain historical information at the beginning stage, and this may lead to poor convergence.

Recently, there are some works exploiting knowledge reuse techniques or machine learning in evolutionary computation that have been proposed. Iqbal *et al.* [34] proposed an approach based on transfer learning and genetic programming to solve complex image classification problems. The basic idea of the proposed algorithm is that the knowledge learned from a simpler subtask is used to solve a more complex subtask, and reusing knowledge blocks are discovered from similar as well as different image classification tasks during the evolutionary process. Iqbal *et al.* [35] presented a genetic programming-like representation to identify building blocks of knowledge in a learning classifier system, and the proposed method can extract useful building blocks from simpler and smaller problems and reuse them to learn more complex multiplexer problem. Feng *et al.* [36] presented an evolutionary memetic computing paradigm that is capable of learning and evolving knowledge meme that traverses two different but related problem domains, capacitated vehicle routing problem and capacitated arc routing problem, for greater search efficiency. Experimental results show that evolutionary optimization can benefit from this approach.

However, we are not exactly sure if the data we using to construct the prediction model and the data we are going to predict by the above model obey a similar distribution. Conversely, the real-world applications repeatedly reminded us that, it is not wise to assume the IID hypothesis as a prerequisite, especially for the DMOPs. Unfortunately, most of the existing methods often assume that the solutions at different times have an IID structure, and we believe that this assumption is one of main reasons for the failure of existing DMOEA algorithms. After all, a poor prediction model is very likely to bring the search process to a hopeless place, which means actual results will be worse than a method which does not use predictive technique, if the prediction model turn out to be inaccurate.

We believe that historical information about the POF or POS is very useful, and the reason is that for a DMOP, the POSs or POFs at different times may not be exactly the same, but they must be correlated. Therefore we conjecture that an appropriate use of the information extracted from the obtained POS or POF will bring great benefits to track the changing POF, but at the same time, we must admit the rationality and the generality of the assumption of nonindependently and identically distributed data. From these basic points of view, we propose a framework which integrates transfer learning and EAs for solving the DMOPs. Two of the major advantages of the proposed approach are as follows: at first, the proposed method does not assume IID hypothesis as a prerequisite, and it is enabled to escape serious consequences of an unsuitable model. Second, this approach is designed to generate a population-building prediction model, so that any population-based optimization algorithms may benefit from this integration without any extensive modification.

III. Transfer Learning-Based Dynamic Multiobjective Optimization Algorithm

In this section, we propose a transfer learning-based dynamic optimization algorithm. Our motivation is that the solutions of a dynamic optimization problem under different environments obey different probability distributions, and these distributions are not identical but are correlated. If we can map these different distributions into a latent space, and in this space the distributions are as "similar" as possible, then we can use the available solutions to generate an initial population, such that the solutions under a new environment can be computed with low computational cost. In principle, this design is a reuse process of the knowledge we already obtained.

Before giving the details of the proposed approach, we need to introduce background information of the DAL we will use it in our design.

A. Transfer Component Analysis

Briefly speaking, DAL [37]–[39], a branch of transfer learning, is to reuse the knowledge acquired from a source domain

to perform a task in a target domain, which is related to, but distinct from the source domain. In the context of this paper, a domain includes a sample space $\mathcal{X}$ and the corresponding marginal distribution $P(X)$, where $X = \{x_1, x_2, \ldots, x_n\} \subseteq \mathcal{X}$. We say that two domains are different, which means they have different sample spaces and the marginal distributions are different.

The researchers [40] believe that it is a promising solution using the DAL to find a goodİ representation to decrease the difference between the distributions of source and target domains. Gretton *et al.* [41] noted that the distance between two different distributions can be evaluated by a particular function, and in the reproducing kernel Hilbert space (RKHS), the computational cost of the evaluation can be reduced. Based on this observation, Gretton *et al.* [41] and Smola *et al.* [42] proposed a nonparametric distance estimation method called maximum mean discrepancy (MMD) to differentiate distributions in the RKHS [43]. The MMD measures the discrepancy between two distributions by computing the difference of the mean values for the source domain and target domain. The advantages of the MMD approach is its simplicity and accuracy.

Definition 4 (MMD [41]): Let p and q be two Borel probability measures defined on a domain $\mathcal{X}$; and $X = \{x_1, \ldots, x_m\}$ and $Y = \{y_1, \ldots, y_n\}$ be two observations drawn from p and q, respectively. Let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$, then the MMD can be defined as

$$\mathrm{MMD}(\mathcal{F}, p, q) \coloneqq \sup_{f \in \mathcal{F}} \left(\frac{1}{m} \sum_{i=1}^{m} f(x_i) - \frac{1}{n} \sum_{i=1}^{n} f(y_i) \right).$$

In an RKHS, f can be written as $f(x) = <\phi(x), f>$, where $\phi(x) : \mathcal{X} \to \mathcal{H}$. So the empirical estimate of MMD can be rewritten as

$$\mathrm{MMD}(\mathcal{F}, p, q) \coloneqq \left\| \frac{1}{m} \sum_{i=1}^{m} \phi(x_i) - \frac{1}{n} \sum_{i=1}^{n} \phi(y_i) \right\|_{\mathcal{H}}^2. \tag{2}$$

By using the so-called "kernel trick" [44], we can rewrite (2) as

$$\begin{aligned} \mathrm{MMD}(\mathcal{F}, p, q) \coloneqq \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{tr} \Bigg[\hat{K} \Bigg(& \frac{1}{m \times m} L_{ii} - \frac{1}{m \times n} L_{ij} \\ & - \frac{1}{n \times m} L_{ji} + \frac{1}{n \times n} L_{jj} \Bigg) \Bigg] \\ \coloneqq \mathbf{tr}\left(\hat{K} L \right) & \end{aligned} \tag{3}$$

where $\mathbf{tr}(A)$ refers to the trace of the matrix A, and the matrix

$$\hat{K} = \begin{pmatrix} \hat{K}_{X,X} & \hat{K}_{X,Y} \\ \hat{K}_{Y,X} & \hat{K}_{Y,Y} \end{pmatrix} \in \mathbb{R}^{(m+n) \times (m+n)}. \tag{4}$$

$\hat{K}_{X,Y}$ is a kernel matrix with $k_{i,j} = \kappa(x_i, y_j) = \phi(x_i)^T \phi(y_j)$, where $\kappa(\cdot, \cdot)$ is a kernel function and $\phi(\cdot)$ is a feature mapping function. This matrix reflects data similarity in the domains X and Y. $\hat{K}_{X,X}$, $\hat{K}_{Y,X}$, and $\hat{K}_{Y,Y}$ have the similar meanings. Matrix L contains the coefficients to scale matrix according to (2) and its elements are as follows:

$$L(i,j) = \begin{cases} \frac{1}{m \times m}, & x_i, x_j \in X \\ \frac{1}{n \times n}, & x_i, x_j \in Y \\ -\frac{1}{m \times n}, & \text{otherwise.} \end{cases} \tag{5}$$

On the basis of the MMD, Pan *et al*. [10] proposed a dimension reduction method called MMD embedding (MMDE) to: 1) find a low-dimensional space to reduce the difference between source and targets distributions and 2) to preserve the main statistical properties, maximization of data variance in the first extracted orthogonal components of the original data X and Y. In MMDE, the kernel function κ is learned (or optimized) from the data, which makes it computationally expensive, so the authors in [10] proposed other dimension reduction-based methods called TCA and its semisupervised version of TCA to transform the problem of learning an entire kernel matrix to a low-rank matrix W instead.

Now let us consider how to obtain the matrix W by using the TCA method. Suppose that W is a $(m + n) \times d$ matrix. For any vector x, let $\phi(x) = W^T \kappa_x \in \mathbb{R}^d$, where $\phi(\cdot)$ is a feature mapping function. Let $\kappa_x = [\kappa(x_1, x), \ldots, \kappa(x_m, x), \kappa(y_1, x), \ldots, \kappa(y_n, x)]^T$, and the matrix $\hat{K}$ in (4) can be transformed as follows:

$$\begin{aligned} \hat{K} &= \left[\phi(x_1), \ldots, \phi(x_m), \phi(y_1), \ldots, \phi(y_n) \right]^T \\ &\quad \times \left[\phi(x_1), \ldots, \phi(x_m), \phi(y_1), \ldots, \phi(y_n) \right] \\ &= \left[W^T \kappa_{x_1}, \ldots, W^T \kappa_{x_m}, W^T \kappa_{y_1}, \ldots, W^T \kappa_{y_n} \right]^T \\ &\quad \times \left[W^T \kappa_{x_1}, \ldots, W^T \kappa_{x_m}, W^T \kappa_{y_1}, \ldots, W^T \kappa_{y_n} \right] \\ &= \left[\kappa_{x_1}, \ldots, \kappa_{x_m}, \kappa_{y_1}, \ldots, \kappa_{y_n} \right]^T W W^T \\ &\quad \times \left[\kappa_{x_1}, \ldots, \kappa_{x_m}, \kappa_{y_1}, \ldots, \kappa_{y_n} \right] \\ &= K^T W W^T K \\ &= K W W^T K. \end{aligned} \tag{6}$$

Please note that the matrix K is a symmetric matrix, so $K^T = K$, and then $\mathbf{tr}(\hat{K}L) = \mathbf{tr}(KWW^TKL)$. According to the property of the trace of a matrix, we can rewrite (3) as follows:

$$\begin{aligned} \mathrm{MMD}(\mathcal{F}, p, q) &= \mathbf{tr}\left(\hat{K} L \right) \\ &= \mathbf{tr}\left(K W W^T K L \right) \\ &= \mathbf{tr}\left(W^T K L K W \right). \end{aligned} \tag{7}$$

Now the optimization problem for the TCA algorithm can be written as follows:

$$\begin{aligned} & \arg\min_{W} \ \mu \cdot \mathrm{tr}(W^T W) + \mathrm{tr}\left(W^T K L K W \right) \\ & \text{subject to} \ \ W^T K H K W = \mathbf{I} \end{aligned} \tag{8}$$

where $H = \mathbf{I} - (1/[m+n])\mathbf{1}\mathbf{1}^T$ and $\mathbf{I}$ is a $(m+n) \times (m+n)$ identity matrix. $W^T W$ is a regularization term. $\mathbf{1}$ is a $(m+n) \times 1$ all-ones matrix. m and n represent the numbers of samples in the source and target domains, respectively. μ is the tradeoff parameter. This optimization problem can be transformed into a trace maximization problem. According to the method presented in [45], the trace maximization problem can be solved by the generalized eigenvalue decomposition, and the solution is composed of the d leading eigenvectors. The pseudo-code of TCA is given in Algorithm 1.

Algorithm 1: TCA

Input: Source domain X; target domain Y; a kernel function $\kappa(\cdot,\cdot)$;
Output: Matrix W

Construct the Kernel Matrix $\hat{K}$, the Matrix L, and the Matrix H according to (4), (5) and (8) ;
Construct the Matrix W by using the d leading eigenvectors of $(KLK + \mu I)^{-1}KHK$;
return the matrix W;

B. Tr-DMOEA

DMOP is a computationally expensive task. This implies that it requires a lot of computational resources to search for the varying POS at a certain time. If the knowledge about the POS and POF can be reused to predict future POFs or POSs under different environments, this usually implies performance improvement as well as less computational resource consumption. As a result, we believe that the prediction-based dynamic multiobjective optimization algorithm presents a promising solution.

However, the existing algorithms generally neglect the assumption of nonindependent identically distributed (Non-IID), and it is obvious that the individuals under different environments obey different distributions. This also means that those dynamic optimization algorithms based on the traditional machine learning approach leave much room for improvement. So we put forward the use of the domain adaptation technique to develop a novel DMOEA.

The approach developed is to map different distributions that the solutions obey at different times into a new latent space via the domain adaptation method. In the latent space, the MMD value of different distributions will be as small as possible while variance of the data will be kept the same. In other words, we will make those distributions that the solutions under different environments obey as similar as possible in the latent space, so we can map the POF we have obtained into the space, and then use those mapped solutions to construct a population which will be used to search for the POF under a new environment.

In the following, Tr-DMOEA algorithm (see pseudo-codes in Algorithm 3), F_t is the current dynamic optimization function assuming its POF has already been found. F_{t+1} is the optimization function at the next time. The major part of the algorithm, transfer learning-based initial population generator (Tr-IPG), utilizes the POF at time t and the transfer learning method to generate a population which can be used to search for the POF at time $t+1$. More specifically, we take the obtained POF at time t as a source domain; the feasible solutions of the next time, time $t+1$, as the target domain, and then construct a mapping function φ by using the domain adaptation approach. This mapping function will embed the distributions that the source and target domain obey separately into a latent space, and in that space the difference between the two distributions will become as small as possible. From this, we can use the POF already found to generate an initial population which can be used to search for the POF of the next moment.

Algorithm 2: Tr-IPG

Input: The Dynamic Optimization Function $F_{t+1}(\cdot)$; the POF of the function $F_t(\cdot)$ at time t, $POF_t = \{p_1, \ldots p_m\}$; a kernel function $\kappa(\cdot,\cdot)$.
Output: A population **Pop-init**.

1 Initialization;
2 For the optimization functions $F_t(\cdot)$ and $F_{t+1}(\cdot)$, randomly generate two sets of the solutions X_s and Y_t ; `/* Remark 1 */`
3 Calculate the objective values of the optimization functions $F_t(X_s)$ and $F_{t+1}(Y_t)$;
4 W $\leftarrow$ TCA($\{F_t(X_s)\}$, $\{F_{t+1}(Y_t)\}$, κ);
5 $PLS \leftarrow \emptyset$; `/* Remark 2 */`
6 **for** *every* $p \in POF_t$ **do**
7 $\quad \kappa_p \leftarrow \left[\kappa(F_t(X_s(1)), p), \ldots, \kappa(F_{t+1}(Y_t(n_t)), p)\right]^T$
8 $\quad \varphi(p) \leftarrow W^T\kappa_p$;
9 $\quad PLS = PLS \cup \{\varphi(p)\}$;
10 **end**
11 **for** *every* $l \in PLS$ **do**
12 $\quad x \leftarrow \operatorname{argmin}_x \|\varphi(F_{t+1}(x)) - l\|$ `/* Remark3 */`
13 $\quad$ **Pop-init = Pop-init** $\cup\{x\}$;
14 **end**
15 **return Pop-init ;**

In order to help the readers quickly grasp the basic idea of the algorithm Tr-IPG (see pseudo-codes in Algorithm 2), Fig. 1 has been presented to illustrate the key elements of the algorithm. This diagram describes the operational process from lines 6 to 15 of the Tr-IPG.

Please note that the input to the TCA are the samples from the solutions at time t and $t+1$, and its output is a transformation matrix W. We can use the matrix W to construct the latent space. Step 1 (i.e., the top left corner of the diagram) depicts the process of mapping the POF at time t into the latent space, and steps 2 and 3 describe how to search for an initial population which can be used to solve the dynamic optimization at time $t+1$.

Remark 1: The numbers of the elements of X_s and Y_t are predefined. Let $|X_s| = n_s$ and $|Y_t| = n_t$. In general, more sampling often means a better result, but it also needs to pay a higher computational cost, so the decision about how many solutions needed to be produced in this step depends on the resources available.

Remark 2: Particle in the latent space (PLS) can be regarded as a set of the mapped solutions in the latent space.

Remark 3: We want to find a decision variable x, such that in the latent space, $\varphi(F_{t+1}(x))$ is closet to $l \in$ PLS in the latent space. This also means that we need to solve a single objective optimization problem here, and any single objective optimization algorithm can be applied to solve the problem. In this paper, we use the interior point algorithm to solve the problem.

What the Tr-IPG algorithm outputs is a population, so it is not difficult to find that we can combine any type of population-based optimization algorithms with the Tr-IPG to obtain a Tr-DMOEA.

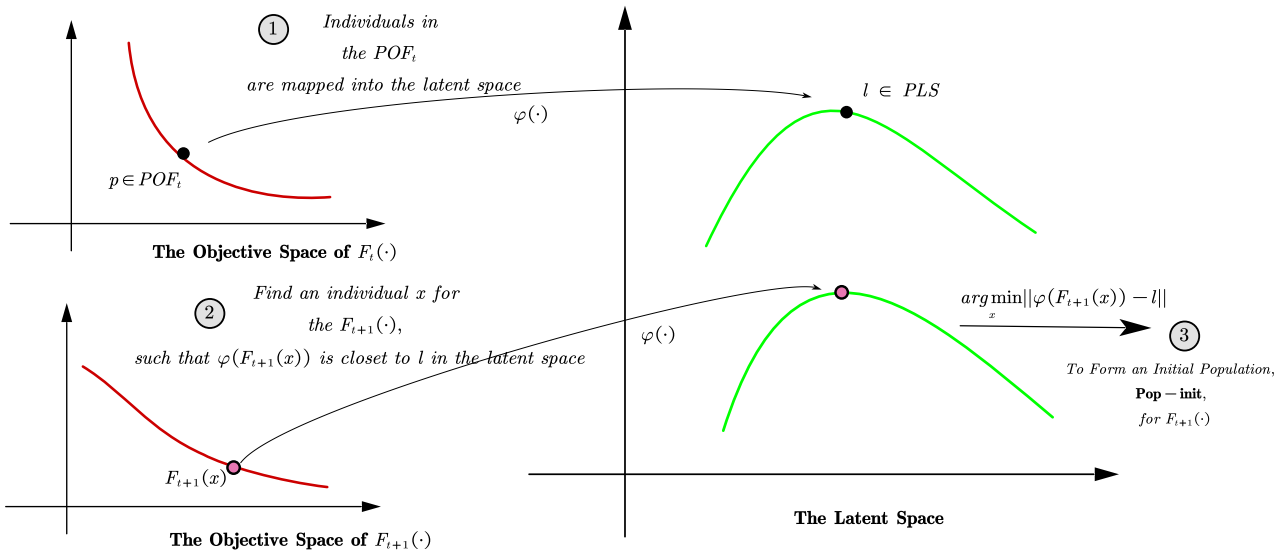


Fig. 1. Key steps of the Tr-IPG algorithm. Step 1: map the obtained POF into the latent space. Step 2: find individuals for $F_{t+1}(\cdot)$. Step 3: generate initial population pool for the problem $F_{t+1}(\cdot)$.

Remark 4: For the TCA algorithm, the major time is spent on eigenvalue decomposition. It takes $O(d(m_1+m_2)^2)$ time when d nonzero eigenvectors are to be extracted, where m_1 and m_2 are the numbers of the solutions which are generated to construct the latent space. The Tr-IPG spends $O(n_1)$ time to map an individuals in the POF_t to the latent space and we use the interior point algorithm to find an individual in the objective space of $F_{t+1}(\cdot)$. For the primal dual interior point method, suppose the constraint matrix A has n rows and m columns, and $n < m$, it has $O(\sqrt{m}L)$ iterations and $O(m^3L)$ arithmetic operations, where L is total number of bits of the input.

Example 1: A specific numerical example is helpful in understanding how the Tr-IPG algorithm works. For example, we employ the Tr-IPG algorithm to solve the FDA4 problem, which is a three-objective dynamic optimization problem. Let us suppose that the POF of the FDA4 problem [46] at time t, POF_t, has been found and $p \in \text{POF}_t$. The Tr-IPG uses the TCA algorithm to obtain a mapping function $\varphi(\cdot)$, and this mapping function is utilized to map the p into a 20-D latent space,[2] and it means that $l = \varphi(p)$ is a 20-D vector. After that the Tr-IPG algorithm will find a solution x for the FDA4 problem at time $t+1$, and this solution x satisfies the requirement that it is nearest to l in the latent space. The Tr-IPG will output the solution x as one of the individuals of the initial population which can be used to solve the FDA4 problem at time $t+1$.

IV. Empirical Study

Practically speaking, the proposed approach is compatible with any type of population-based optimization algorithms. As a case study, in our experiments, we select three well-represented algorithms with different operating metaphors to verify our approach. The first one is the NSGA-II [47] and it is a multiobjective genetic algorithm that applies nondominated sorting and crowding distance. The second multiobjective optimization algorithm is based on PSO and it is simply called as MOPSO [48]. The third one is the RM-MEDA [49], which is a regularity model-based multiobjective estimation of distribution algorithm.

The three corresponding algorithms with the proposed transfer learning are called Tr-NSGA-II, Tr-MOPSO, and Tr-RM-MEDA, respectively, for dynamic optimization. It should be noted that the original designs, NSGA-II, MOPSO, and RM-MEDA are not appropriate for dynamic optimization. It is not difficult to find that these three algorithms belong to different categories, but all of them are well-developed, so it can strengthen the persuasive power and the confidence level to incorporate the proposed technology. At the same time, we also compare the new algorithms with other state-of-art designs.

Algorithm 3: Tr-DMOEA

Input: The Dynamic Optimization Function $F(X)$; a population based multiobjective algorithm MOA; a kernel function $\kappa(\cdot,\cdot)$.
Output: the POFs of $F(X)$.

```
Initialization ;
Use MOA to solve F_0(X) to get a POF_0 ;
for t = 1 to n do
    Next-Pop = Tr-IPG(F_t(·), POF_(t−1), κ(·,·)) ;
    /* When a change occurred, we use
    Tr-IPG to generate an init
    population. */
    POF_t = MOA(Next-Pop) ;
    return POF_t ;
end
```

[2]The dimensionality of the latent space depends on the parameters of the TCA algorithm.

One thing we need to emphasize is that in all our experiments, the parameters are set the same. In other words, for these 12 test functions and three different algorithms, we have used the same parameters and do not tune the parameters in TCA under different configurations for a better performance.

A. Performance Metrics, Testing Functions, and Settings

In this paper, we use four performance metrics, the inverted generational distance (IGD) and its variants, the reactivity measure (React) and its variants, to evaluate the quality of the solutions obtained by these competing algorithms.

1) The IGD [50] is a metric to quantify the performance of a multiobjective optimization algorithm. Let P^* be the set of uniformly distributed Pareto optimal solutions in the POF and P represent the POF obtained by the algorithm, the definition of the IGD is

$$\mathrm{IGD}\big(P^*, P, C\big) = \frac{\sum_{v^* \in P^*} \min_{v \in P} \|v^* - v\|}{|P^*|}. \quad (9)$$

If we want the value of IGD to be as small as possible, the P should be close enough to P^*. In other words, the IGD depicts the difference between the ideal POF and the POF obtained by the competing algorithms. Please note that the definition of the IGD is slightly different from the original one, and the major difference is the parameter C in (9). The parameter C is a combination of the benchmark functions parameters. We call it as configuration of the benchmark functions. The configurations we used in our experiments are described in Table II.

2) One variant of the IGD, called MIGD, can also be used to evaluate dynamic multiobjective optimization algorithms [51], [52], and it takes the average of the IGD values in some time steps over a run as the performance metric, given by

$$\mathrm{MIGD}\big(P^*, P, C\big) = \frac{1}{|T|} \sum_{t \in T} \mathrm{IGD}\big(P_t^*, P_t, C\big) \quad (10)$$

where P_t^* and P^t represent the points set of the ideal POF and the approximate POF obtained by the algorithm at time t. We also want to evaluate those algorithms under different environments, so a novel metric, DMIGD, is defined based on the MIGD, and the definition of the DMIGD is as follows:

$$\mathrm{DMIGD}\big(P^*, P, C\big) = \frac{1}{|E|} \sum_{C \in E} \mathrm{MIGD}\big(P_t^*, P_t, C\big) \quad (11)$$

where $|E|$ is the number of the different environments experienced. In our experiments, we choose eight different configurations. As a result, $|E|$ equals to eight. What we want to point out is that the DMIGD can evaluate a dynamic optimization algorithm from a high-level view and it bears a significant difference with the MIGD since the MIGD just considers the dynamics in one environment.

3) The React [53] is used to measure the robustness of an algorithm, and its definition is as follows:

$$\mathrm{React}_\epsilon(t, C) = \min\left\{ t' - t | t < t' \in \mathbb{N}, \frac{\mathrm{acc}(t')}{\mathrm{acc}(t)} \geq 1 - \varepsilon \right\}$$

where $\mathrm{acc}(t) = ([\mathrm{HV}(\mathrm{POF}(t))]/[\max \mathrm{HV}(\mathrm{POF})])$ implies the accuracy rate of computing the POF at time t, and HV refers to the value of hypervolume [54]. The React describes how quickly a dynamic optimization algorithm can recover from a change, or convergence speed after changes. The value of the React is the smaller the better. We also want to evaluate the algorithms on a macro-scale, so we derive two additional metrics based on the React

$$\mathrm{MReact}_\epsilon(T, C) = \frac{1}{|T|} \sum_{t \in T} \mathrm{React}_\epsilon(t, C)$$

$$\mathrm{DMReact}_\epsilon(T, C) = \frac{1}{|E|} \sum_{C \in E} \mathrm{MReact}_\epsilon(T, C). \quad (12)$$

The MReact value can be considered as an average of the React values at different time points, but under the same configuration. DMReact is an average of the MReact values over different configurations considered.

In the experiments, we apply the IEEE CEC 2015 Benchmark problems set in Table I as test functions and the problem set has 12 testing functions [46]. In the definitions, the decision variables are $x = (x_1, \ldots, x_n)$ and $t = (1/n_t)\lfloor(\tau_T/\tau_t)\rfloor$, where n_t, τ_T, and τ_t are the severity of change, maximum number of iterations, and frequency of change, respectively. Table II describes the different combinations of n_t, τ_t, and τ_T. Please note that for each n_t-τ_T combination, there will be (τ_T/τ_t) environment changes. In other words, in all of our experiments, there are altogether twenty changes for the 12 dynamic problems.

The POFs of the testing functions have different shapes and each function belongs to a certain DMOPs type. Fig. 2 describes the true POFs of the 12 testing functions and we let the functions change three times. Type I implies POS changes, but POF does not change. Type II means that the POS and the POF change as well. Type III refers to the condition where the POF changes, but the POS does not change. From Table I, we can find that the POF of the functions could be nonconvex, convex, isolated, deceptive, continuous or discontinuous. FDA4 and FDA5 are three-objective functions while all of the remaining are two-objective functions.

The dimensions of the decision variables are from 10 to 30 dimensions. Please note that the A, B, and C values for the functions $\mathrm{FDA5_{iso}}$, $\mathrm{FDA_{dec}}$, $\mathrm{DMOP2_{iso}}$, and $\mathrm{DMOP2_{dec}}$ are set to $G(t)$, 0.001 and 0.05, respectively.

In all of the experiments, we set the population size to 200 and in each generation every algorithm will generate no more than 200 solutions. As mentioned above, we force each benchmark function to change 20 times, and in every change, we let the population carry out 50 iterations.

For the TCA parameters, we set the Gaussian kernel function to the default value and the expected dimensionality was set to be 20. The value of μ was set to 0.5.

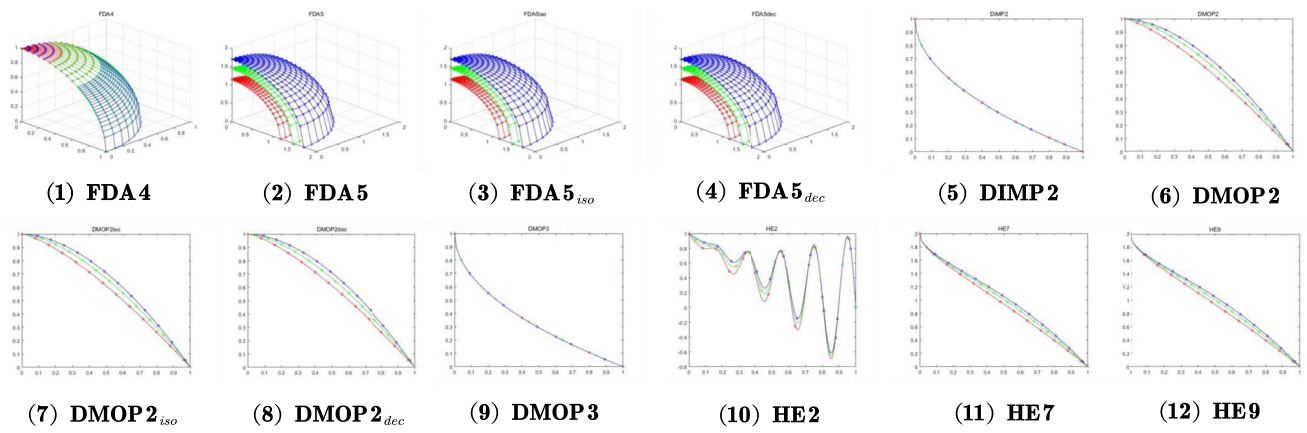


Fig. 2. True POFs of the 12 testing functions. The red, green, and blue lines depict the true POFs at the time steps zero, one, and two, respectively.

TABLE I
BENCHMARK FUNCTIONS

Name	Dim.	# of Obj.	Type
FDA4	12	3	TYPE I
FDA5	12	3	TYPE II
FDA5$_{iso}$	12	3	TYPE II
FDA5$_{dec}$	12	3	TYPE II
DIMP2	10	2	TYPE I
DMOP2	10	2	TYPE II
DMOP2$_{iso}$	10	2	TYPE II
DMOP2$_{dec}$	10	2	TYPE II
DMOP3	10	2	TYPE I
HE2	30	2	TYPE III
HE7	10	2	TYPE III
HE9	10	2	TYPE III

TABLE II
CONFIGURATIONS OF THE BENCHMARK FUNCTIONS PARAMETERS

	n_t	T_t	T_T
C1	10	5	100
C2	10	10	200
C3	10	25	500
C4	10	50	1000
C5	1	10	200
C6	1	50	1000
C7	20	10	200
C8	20	50	1000

B. Experimental Results

1) IGD Metric: For each benchmark function, we perform tests under eight different configurations which are listed in Table II. Each function will change 20 times, and after each change the algorithms would return a POS, and then we calculated the IGD, MIGD, and DMIGD values, respectively.

The detailed results of these experiments are described in 12 tables, and supplemental material contains those tables. These tables recorded the MIGD values of the algorithms running on different testing functions under different environments. In these tables, the "ROC" refers to the ratio of change of the MIGD values and we used bold face to identify those experiments where performance has been improved.

For the convenience of the readers and at the same time owing to space constraints, we summarize these experimental results in three tables, Tables III–V. These three tables illustrate the rate of change of the three new algorithms compared to their original designs. For example, the value of the first row (under C1 configuration) and the first column (under FDA4) of Table III is 61.32, and this shows that the Tr-NSGA-II algorithm improves the NSGA-II algorithm by 61.32% when dealing with the FDA4 problem under the configuration C1.

Please note that our experimental results are obtained without explicitly tuning the parameters one by one, and if we adjust the parameters separately for different algorithms, we have reason to believe that we can get better experimental results. The reason that we did not tune the parameters specifically for getting better results is that the 12 test functions are not exactly the same, so we can set different parameters to obtain the best performance for each test function. For example, we can construct different latent spaces for the 12 benchmark functions individually via setting different parameters of the TCA method. However, we think that this one-by-one-adjustment strategy does not effectively explain the advantages of our approach since almost all algorithms can obtain a better performance via such parameter-tuning, and this makes no contribution to explain the superiority of the proposed algorithm.

We tabulate all the experimental results and obtain the following observations. For the NSGA-II, the overall effective rate of the Tr-NSGA-II was 78% (i.e., 75 cases with improving performance out of 96 total tests), of which 33 testing cases increased by more than 50%, 38 increased by 5%–50% and four cases improved by 0%–5%; for the MOPSO, the total effective rate was 70% (i.e., 67 cases with improving performance out of 96 total tests), including 29 testing cases improved by more than 50%, 33 performance improved by 5%–50%, and five improved by 0%–5%. For the RM-MEDA, the total effective rate was 73% (i.e., 70 cases with improving performance out of 96 total tests), including 23 of the test cases increasing by more than 50%, 39 lifting 5%–50%, and eight improved by 0%–5%.

These experimental results demonstrate that the transfer learning technique can improve the performance of

TABLE III
RATIO OF CHANGE OF MIGD VALUE BETWEEN TR-NSGA-II AND NSGA-II

ROC(%)	C1	C2	C3	C4	C5	C6	C7	C8
FDA4	**61.32**	**62.15**	**59.42**	**59.60**	**78.57**	**79.75**	**56.17**	**61.02**
FDA5	**41.13**	**37.80**	-14.36	**14.45**	**37.12**	**39.04**	**20.39**	**33.09**
$FDA5_{iso}$	-48.91	-17.37	-31.83	-31.98	-20.14	-28.66	-2.44	-6.15
$FDA5_{dec}$	**19.83**	**17.09**	**26.06**	**40.23**	**22.52**	**3.07**	**51.88**	**41.34**
DIMP2	**30.70**	**42.41**	**39.57**	**34.61**	**47.30**	**38.69**	**37.57**	**44.54**
DMOP2	**74.97**	**66.52**	**78.43**	**67.01**	-676.29	**88.96**	**61.22**	**72.65**
$DMOP2_{iso}$	-4.84	-4.46	-6.81	-3.18	-7.54	-13.79	-2.98	-2.48
$DMOP2_{dec}$	**44.91**	**51.25**	**44.39**	**40.13**	**23.26**	**42.45**	**45.66**	**25.58**
DMOP3	**72.70**	**73.61**	**83.50**	**57.65**	-514.50	**4.32**	**74.18**	**76.29**
HE2	**2.73**	-17.45	**1.67**	**5.86**	**67.45**	**77.01**	**28.07**	-19.57
HE7	**57.10**	**58.57**	**59.45**	**61.57**	**53.47**	**58.34**	**57.77**	**61.98**
HE9	**14.79**	**13.02**	**16.64**	**12.99**	**14.98**	**14.61**	**16.14**	**15.49**

TABLE IV
RATIO OF CHANGE OF MIGD VALUE BETWEEN TR-MOPSO AND MOPSO

ROC(%)	C1	C2	C3	C4	C5	C6	C7	C8
FDA4	**18.01**	**16.44**	**15.85**	**17.34**	**14.15**	**13.09**	**18.98**	**18.88**
FDA5	**43.09**	**56.21**	**25.44**	**21.56**	**60.00**	**57.88**	**29.65**	**23.78**
$FDA5_{iso}$	-11.03	-7.58	-2.12	-1.49	-12.51	-5.11	-6.98	-8.00
$FDA5_{dec}$	**17.54**	**30.66**	**50.61**	**23.97**	**10.64**	**11.78**	**28.18**	**20.68**
DIMP2	**86.54**	**88.36**	**93.05**	**94.73**	**73.76**	**87.17**	**83.68**	**89.37**
DMOP2	**92.85**	**96.77**	**96.51**	**94.93**	-478.53	**78.24**	**26.14**	**84.36**
$DMOP2_{iso}$	-0.26	**2.55**	**1.64**	**2.82**	-0.01	-0.04	-0.58	**3.31**
$DMOP2_{dec}$	**89.96**	**93.98**	**93.92**	**96.53**	**92.54**	**92.10**	**97.14**	**96.66**
DMOP3	-296.04	**54.28**	**19.34**	**28.14**	-1371.99	**94.39**	**51.00**	**26.72**
HE2	**14.04**	**34.03**	**11.11**	**34.34**	**38.46**	**36.01**	-1.96	**14.06**
HE7	**2.88**	-3.40	-1.78	-0.49	**13.96**	**8.11**	-14.43	-9.12
HE9	-11.81	-11.45	-12.29	-14.04	-37.00	-33.30	-16.15	-13.57

the existing multiobjective EAs appreciably without significant modifications for solving the DMOPs. On the other hand, we would like to point out that most of the testing cases of performance degradation came from two functions: 1) $FDA5_{iso}$ and 2) $DMOP2_{iso}$. The common characteristic of these two functions is the isolated POFs, so we suspect that the reason why the performance is poor for the two benchmark functions is inappropriate parameters settings.

We also compare the DMIGD value with some chosen state-of-the-art designs, including multidimensional Bayesian network-based estimation distribution algorithm (MBN-EDA) [55], random immigrants strategy-based multiobjective differential EA with decomposition (RND), and the Kalman

TABLE V
RATIO OF CHANGE MIGD VALUE BETWEEN TR-RM-MEDA AND RM-MEDA

ROC(%)	C1	C2	C3	C4	C5	C6	C7	C8
FDA4	**21.47**	**22.74**	**22.42**	**22.91**	**25.75**	**27.71**	**22.32**	**23.08**
FDA5	**54.07**	**59.96**	**61.66**	**57.40**	**76.55**	**69.49**	**56.24**	**54.71**
$FDA5_{iso}$	-0.29	-0.09	0.08	-0.30	1.81	-0.06	-0.63	-0.08
$FDA5_{dec}$	**39.69**	**33.15**	**37.16**	**35.83**	**65.97**	**65.59**	**41.72**	**45.66**
DIMP2	**5.91**	-8.53	-2.03	-3.49	-1.08	**1.01**	-2.33	-4.57
DMOP2	-827.79	-52.71	**37.53**	**33.36**	-3.34	-1.67	**21.07**	-13.04
$DMOP2_{iso}$	-0.01	-0.09	-0.05	-0.06	**0.01**	-0.01	**0.07**	-0.09
$DMOP2_{dec}$	**53.72**	**64.32**	**51.05**	**60.02**	**1.39**	**3.85**	**57.93**	**49.28**
DMOP3	**24.65**	**22.08**	-25.15	**0.49**	-0.45	-0.79	**10.75**	**27.48**
HE2	**86.81**	**87.01**	**86.74**	**88.01**	**89.99**	**89.59**	**88.42**	**87.91**
HE7	**13.31**	**21.82**	**21.40**	**20.29**	**19.23**	**14.87**	**21.32**	**22.57**
HE9	**7.48**	**7.61**	**6.95**	**6.91**	**8.35**	**7.53**	**6.74**	**5.43**

TABLE VI
DMIGD VALUES OF DIFFERENT ALGORITHMS

DMIGD	NSGA-II	Tr-NSGA-II	MOPSO	Tr-MOPSO	RM-MEDA	Tr-RM-MEDA	MBN-EDA	RND	MOEA/D-KF
FDA4	0.2634	0.0858	0.0732	0.0609	0.0680	**0.0520**	0.43	0.1698	0.1913
FDA5	0.3301	0.2306	0.2131	0.1196	0.2089	**0.0776**	0.51	0.5323	0.4963
$FDA5_{iso}$	0.1048	0.1292	0.1106	0.1181	0.0650	**0.0649**	0.64	0.1433	0.1465
$FDA5_{dec}$	0.5923	0.4559	0.2746	0.2139	0.5779	**0.3275**	1.27	0.5403	0.5476
DIMP2	3.8986	2.3502	2.3684	**0.2937**	4.8892	4.9769	6.97	17.9537	22.9536
DMOP2	0.4202	0.3439	**0.2129**	0.2538	4.5942	4.7130	1.4	1.4329	3.0619
$DMOP2_{iso}$	0.0325	0.0358	0.0319	0.0318	**0.0290**	**0.0290**	2.56	0.0315	0.0316
$DMOP2_{dec}$	0.6303	0.3930	0.4192	**0.0254**	0.1449	0.0940	2.89	9.0504	9.2188
DMOP3	0.8851	1.0133	0.2851	**0.2650**	4.5897	4.6177	1.38	0.0697	0.0836
HE2	0.2096	0.1501	0.0847	**0.0640**	0.8451	0.1017	0.83	0.0744	0.0745
HE7	0.0946	0.0390	0.0582	0.0583	0.0428	**0.0342**	0.21	0.1787	0.2365
HE9	0.2954	0.2508	0.2459	0.2887	0.2565	**0.2383**	0.36	0.3432	0.4108

filter prediction-based DMOEA (MOEA/D-KF) [51], [52], and the results are depicted in Table VI. The experimental results show that the transfer learning-based algorithms have much better performance over different problem characteristics in these benchmark functions. Even compared with chosen state-of-the-art algorithms, these transfer learning-based algorithms can be much more efficient.

2) React Metric: Table VII depicts the DMReact values of all competing algorithms. We found that the Tr-NSGA-II and Tr-MOPSO have shown improvements, which suggest that, at least at this parameters setting, the proposed approach can improve the adaptability of the NSGA-II and the MOPSO under dynamic environments. However, the robustness of the RM-MEDA seems to be reduced, and one reason we envision is that the TCA method have coincidentally reduced the

TABLE VII
DMREACT VALUE OF THE ALGORITHMS

DMREACT	NSGA-II	Tr-NSGA-II	MOPSO	Tr-MOPSO	RM-MEDA	Tr-RM-MEDA
FDA4	1.9803	**1.7664**	1.4934	**1.2895**	1.5033	1.7072
FDA5	1.7204	**1.5625**	1.4375	**1.3092**	2.9638	**1.6086**
$FDA5_{iso}$	1.7105	**1.4539**	1.6283	1.7039	1.0329	**1.0066**
$FDA5_{dec}$	1.7928	1.8224	1.5132	1.9375	2.5000	2.5263
DIMP2	2.2697	2.2763	1.4013	**1.1151**	1.9243	2.0197
DMOP2	2.1645	**1.9671**	1.5592	1.8487	1.5789	1.7467
$DMOP2_{iso}$	1.4375	1.4836	1.4704	**1.3586**	1.3355	1.4605
$DMOP2_{dec}$	2.2961	**2.0164**	1.5461	2.1809	1.9309	2.1316
DMOP3	1.7039	**1.3816**	1.5329	**1.3026**	1.0987	1.1349
HE2	2.0428	**1.9474**	1.0987	1.2862	2.1086	**1.4572**
HE7	1.0000	**1.0000**	1.0000	**1.0000**	1.0000	**1.0000**
HE9	1.0000	**1.0000**	1.0000	1.3125	1.0000	**1.0000**

diversity of the solutions. So how to improve the diversity and the robustness at the same time is an interesting topic for the future research. The optimal choice of the parameter setting will be a topic in our future research.

V. CONCLUSION

In this paper, we propose an approach of exploiting a transfer learning technique to enhance the performance of DMOEAs. Our idea is that the solutions of a given DMOP at different times have different distributions, though there are some relationships between these probability distributions, they are not identical. This is a typical Non-IID problem, and classical machine learning methods are difficult to solve it.

For this reason, it is not surprise to understand why the traditional dynamic optimization algorithms designed based on classical machine learning find it hard to achieve satisfactory performance. To overcome these problems, we employ the techniques from the transfer learning to develop an algorithmic framework, which creates benefits for a variety of population-based DMOEAs.

In our approach, we consider different probability distributions that the solutions obey at various times as the source and target domains, respectively. We can exploit the gained POF from the source domain to improve computational efficiency in searching for the POF at the next time instance. To achieve this goal, the transfer learning technique is applied to find a latent space where the global feature, MMD value, of the source and target domains is as small as possible. Meanwhile the major statistical characteristics of the data, i.e., the variance will remain unchanged. In this way, we can use the obtained POS to construct an initial population which can be employed by any population-based optimization algorithms to find the POS of the next time instance.

We applied the proposed idea to improve three well-known multiobjective optimization algorithms: 1) NSGA-II; 2) MOPSO; and 3) RM-MEDA. The enhanced algorithms are compared with the original designs and some chosen competing algorithms on a well-adopted benchmark set which involves 12 testing functions. Almost all the experimental results validate that introducing the transfer leaning technique into the dynamic optimization algorithm can greatly improve the quality of the solutions and robustness of the algorithms. This line of research proposed herein can be regarded as a new avenue for designing effective and efficient EAs for DMOPs. A rich body of machine learning techniques can inspire further innovations in solving real-world application [56] with various degrees of complexities and uncertainties.

ACKNOWLEDGMENT

The authors would like to thank A. Muruganantham for providing the code of [51] and M. Wang for experimental assistance.

REFERENCES

[1] 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, Oct. 2004.

[2] C. Cruz, J. R. González, and D. A. Pelta, "Optimization in dynamic environments: A survey on problems, methods and measures," *Soft Comput.*, vol. 15, no. 7, pp. 1427–1448, Jul. 2011.

[3] J. Xu, P. B. Luh, F. B. White, E. Ni, and K. Kasiviswanathan, "Power portfolio optimization in deregulated electricity markets with risk management," *IEEE Trans. Power Syst.*, vol. 21, no. 4, pp. 1653–1662, Nov. 2006.

[4] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 6, pp. 1–24, Oct. 2012.

[5] M. Daneshyari and G. G. Yen, "Cultural-based particle swarm for dynamic optimisation problems," *Int. J. Syst. Sci.*, vol. 43, no. 7, pp. 1284–1304, 2012.

[6] A. Simões and E. Costa, "Prediction in evolutionary algorithms for dynamic environments," *Soft Comput.*, vol. 18, no. 8, pp. 1471–1497, Aug. 2014.

[7] C. Rossi, M. Abderrahim, and J. C. Díaz, "Tracking moving optima using Kalman-based predictions," *Evol. Comput.*, vol. 16, no. 1, pp. 1–30, Mar. 2008.

[8] P. D. Stroud, "Kalman-extended genetic algorithm for search in nonstationary environments with noisy fitness evaluations," *IEEE Trans. Evol. Comput.*, vol. 5, no. 1, pp. 66–77, Feb. 2001.

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

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

[11] D.-C. Dang, T. Jansen, and P. K. Lehre, "Populations can be essential in dynamic optimisation," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Madrid, Spain, 2015, pp. 1407–1414.

[12] D.-C. Dang, T. Jansen, and P. K. Lehre, "Populations can be essential in tracking dynamic optima," *Algorithmica*, vol. 78, no. 2, pp. 660–680, 2017.

[13] C. Raquel and X. Yao, *Dynamic Multi-Objective Optimization: A Survey of the State-of-the-Art* (Studies in Computational Intelligence). Heidelberg, Germany: Springer, 2013, pp. 85–106.

[14] R. Azzouz, S. Bechikh, and L. B. Said, "Dynamic multi-objective optimization using evolutionary algorithms: A survey," in *Recent Advances in Evolutionary Multi-Objective Optimization*. Cham, Switzerland: Springer, Aug. 2016, pp. 31–70.

[15] C.-K. Goh and K. C. Tan, "Dynamic evolutionary multi-objective optimization," in *Evolutionary Multi-Objective Optimization in Uncertain Environments*. Heidelberg, Germany: Springer, 2009, pp. 125–152.

[16] H. G. Cobb, "An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments," Inf. Technol. Division, U.S. Navy Center Appl. Res. Artif. Intell., Washington, DC, USA, Tech. Rep. AIC-90-001, 1990.

[17] F. Vavak, T. C. Fogarty, and K. Jukes, "A genetic algorithm with variable range of local search for tracking changing environments," in *Parallel Problem Solving From Nature—PPSN IV*. Heidelberg, Germany: Springer, 1996, pp. 376–385.

[18] Y. G. Woldesenbet and G. G. Yen, "Dynamic evolutionary algorithm with variable relocation," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 500–513, Jun. 2009.

[19] J. Grefenstette, "Genetic algorithms for changing environments," in *Parallel Problem Solving From Nature 2*. Amsterdam, The Netherlands: North-Holland, 1992, pp. 137–144.

[20] S. Yang and R. Tinós, "A hybrid immigrants scheme for genetic algorithms in dynamic environments," *Int. J. Autom. Comput.*, vol. 4, no. 3, pp. 243–254, Jul. 2007.

[21] M. Mavrovouniotis and S. Yang, "Genetic algorithms with adaptive immigrants for dynamic environments," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, Jun. 2013, pp. 2130–2137.

[22] S. Yang, "Genetic algorithms with memory-and elitism-based immigrants in dynamic environments," *Evol. Comput.*, vol. 16, no. 3, pp. 385–416, Sep. 2008.

[23] K. Deb, Udaya Bhaskara Rao N., and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling," in *Lecture Notes in Computer Science*. Heidelberg, Germany: Springer, 2007, pp. 803–817.

[24] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," in *Proc. Congr. Evol. Comput.*, Washington, DC, USA, 1999, p. 1882.

[25] C.-K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 103–127, Feb. 2009.

[26] Y. Wang and B. Li, "Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization," *Memetic Comput.*, vol. 2, no. 1, pp. 3–24, Mar. 2010.

[27] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 29–38, Feb. 2006.

[28] R. Azzouz, S. Bechikh, and L. B. Said, "A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy," *Soft Comput.*, vol. 21, no. 4, pp. 885–906, Feb. 2017.

[29] J. Branke, T. Kaussler, C. Smidt, and H. Schmeck, "A multi-population approach to dynamic optimization problems," in *Evolutionary Design and Manufacture*. London, U.K.: Springer, 2000, pp. 299–307.

[30] C. Li and S. Yang, "Fast multi-swarm optimization for dynamic optimization problems," in *Proc. 4th Int. Conf. Nat. Comput.*, Jinan, China, 2008, pp. 624–628.

[31] S. Yang and C. Li, "A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 959–974, Dec. 2010.

[32] P. A. N. Bosman, *Learning and Anticipation in Online Dynamic Optimization* (Studies in Computational Intelligence). Heidelberg, Germany: Springer, 2007, pp. 129–152.

[33] A. Zhou, Y. Jin, and Q. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 1, pp. 40–53, Jan. 2014.

[34] M. Iqbal, B. Xue, H. Al-Sahaf, and M. Zhang, "Cross-domain reuse of extracted knowledge in genetic programming for image classification," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 569–587, Aug. 2017.

[35] M. Iqbal, W. N. Browne, and M. Zhang, "Reusing building blocks of extracted knowledge to solve complex, large-scale Boolean problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 465–480, Aug. 2014.

[36] L. Feng, Y.-S. Ong, M.-H. Lim, and I. W. Tsang, "Memetic search with interdomain learning: A realization between CVRP and CARP," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 644–658, Oct. 2015.

[37] S. Ben-David *et al.*, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 151–175, 2010.

[38] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, "Visual domain adaptation: A survey of recent advances," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 53–69, May 2015.

[39] M. Jiang, W. Huang, Z. Huang, and G. G. Yen, "Integration of global and local metrics for domain adaptation learning via dimensionality reduction," *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 38–51, Jan. 2017.

[40] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 19. 2007, pp. 137–144.

[41] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *Proc. Adv. Neural Inf. Process. Syst.*, 2006, pp. 513–520.

[42] A. Smola, A. Gretton, L. Song, and B. Schölkopf, "A Hilbert space embedding for distributions," in *Algorithmic Learning Theory*. Heidelberg, Germany: Springer, 2007, pp. 13–31.

[43] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," *J. Mach. Learn. Res.*, vol. 2, pp. 67–93, Mar. 2002.

[44] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[45] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers, "Fisher discriminant analysis with kernels," in *Proc. IEEE Signal Process. Soc. Workshop Neural Netw. Signal Process. IX*, Madison, WI, USA, 1999, pp. 23–25.

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

[47] 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, Apr. 2002.

[48] C. A. C. Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. Congr. Evol. Comput. (CEC)*, vol. 2. Honolulu, HI, USA, 2002, pp. 1051–1056.

[49] 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, Feb. 2008.

[50] M. R. Sierra and C. A. C. Coello, "Improving PSO-based multi-objective optimization using crowding, mutation and ϵ-dominance," in *Proc. Int. Conf. Evol. Multi Criterion Optim.*, Guanajuato, Mexico, 2005, pp. 505–519.

[51] A. Muruganantham, K. C. Tan, and P. Vadakkepat, "Solving the IEEE CEC 2015 dynamic benchmark problems using Kalman filter based dynamic multiobjective evolutionary algorithm," in *Proc. Adapt. Learn. Optim.*, Nov. 2015, pp. 239–252.

[52] A. Muruganantham, K. C. Tan, and P. Vadakkepat, "Evolutionary dynamic multiobjective optimization via Kalman filter prediction," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2862–2873, Dec. 2016.

[53] M. C. Sola, "Parallel processing for dynamic multi-objective optimization," Ph.D. dissertation, Dept. Comput. Archit. Comput. Technol., Universidad de Granada, Granada, Spain, 2010.

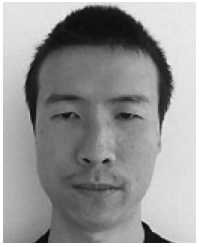
[54] A. J. Nebro *et al.*, "ABYSS: Adapting scatter search to multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 4, pp. 439–457, Aug. 2008.

[55] H. Karshenas, R. Santana, C. Bielza, and P. Larrañaga, "Multiobjective estimation of distribution algorithm based on joint modeling of objectives and variables," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 519–542, Aug. 2014.
[56] Y. Sun, G. G. Yen, and Z. Yi, "Reference line-based estimation of distribution algorithm for many-objective optimization," *Knowl. Based Syst.*, vol. 132, pp. 129–143, Sep. 2017.

Min Jiang (M'11–SM'12) received the bachelor's and Ph.D. degrees in computer science from Wuhan University, Wuhan, China, in 2001 and 2007, respectively.

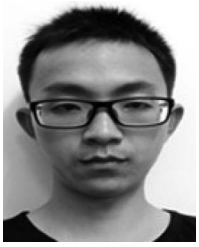
He was a Post-Doctoral Research Fellow with the Department of Mathematics, Xiamen University, Xiamen, China, where he is currently an Associate Professor with the Department of Cognitive Science and Technology. His current research interests include machine learning, computational intelligence, neural-symbolic integration, dynamic multiobjective optimization, machine learning, software development, and basic theories of robotics.

Dr. Jiang was a recipient of the Outstanding Reviewer Award from the IEEE TRANSACTIONS ON CYBERNETICS in 2016. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS. He is the Chair of the IEEE CIS Xiamen Chapter.



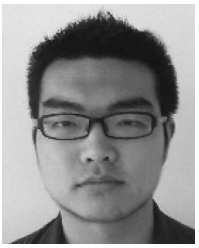
Zhongqiang Huang received the bachelor's and master's degrees in cognitive science and technology from Xiamen University, Xiamen, China, in 2013 and 2016, respectively.

He is currently an Algorithm Engineer with Sangfor Technologies, Shenzhen, China. His current research interests include computational intelligence and machine learning.



Liming Qiu received the bachelor's degree from Fujian Agriculture and Forestry University, Fuzhou, China, in 2015. He is currently pursuing the master's degree with the Department of Cognitive Science and Technology, Xiamen University, Xiamen, China.

He is a member of the Fujian Key Laboratory of the Brain-Like Intelligent Systems, Xiamen University. His current research interests include multiobjective optimization and evolutionary computation.



Wenzhen Huang received the bachelor's degree in cognitive science and technology from Xiamen University, Xiamen, China, in 2014. He is currently pursuing the master's degree with the Institute of Automation, Chinese Academy of Sciences, Beijing, China.

His current research interests include computational intelligence and machine learning.

Gary G. Yen (S'87–M'88–SM'97–F'09) received the Ph.D. degree in electrical and computer engineering from the University of Notre Dame, Notre Dame, IN, USA, in 1992.

He is currently a Regents Professor with the School of Electrical and Computer Engineering, Oklahoma State University (OSU), Stillwater, OK, USA. Before joining OSU in 1997, he was with the Structure Control Division, U.S. Air Force Research Laboratory, Albuquerque, NM, USA. His current research interests include intelligent control, computational intelligence, conditional health monitoring, signal processing, and their industrial/defense applications.

Dr. Yen was a recipient of the Andrew P Sage Best Transactions Paper Award from the IEEE Systems, Man and Cybernetics Society in 2011, and the Meritorious Service Award from the IEEE Computational Intelligence Society in 2014. He was an Associate Editor of the *IEEE Control Systems Magazine*, the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, *Automatica*, *Mechatronics*, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART B: CYBERNETICS, and the IEEE TRANSACTIONS ON NEURAL NETWORKS. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON CYBERNETICS. He was the Founding Editor-in-Chief of the *IEEE Computational Intelligence Magazine* from 2006 to 2009. He served as the General Chair for the 2003 IEEE International Symposium on Intelligent Control held in Houston, TX, USA, and the 2006 IEEE World Congress on Computational Intelligence held in Vancouver, BC, Canada. He served as the Vice President for the Technical Activities in 2005 and 2006 and then the President of the IEEE Computational intelligence Society in 2010 and 2011.