Interactive Evolutionary Multiobjective Optimization via Learning to Rank

Ke Li, Senior Member, IEEE, Guiyu Lai, and Xin Yao, Fellow, IEEE

Abstract—In practical multicriterion decision making, it is cumbersome if a decision maker (DM) is asked to choose among a set of tradeoff alternatives covering the whole Pareto-optimal front. This is a paradox in conventional evolutionary multiobjective optimization (EMO) that always aim to achieve a well balance between convergence and diversity. In essence, the ultimate goal of multiobjective optimization is to help a DM identify solution(s) of interest (SOI) achieving satisfactory tradeoffs among multiple conflicting criteria. Bearing this in mind, this article develops a framework for designing preference-based EMO algorithms to find SOI in an interactive manner. Its core idea is to involve human in the loop of EMO. After every several iterations, the DM is invited to elicit her feedback with regard to a couple of incumbent candidates. By collecting such information, her preference is progressively learned by a learning-to-rank neural network and then applied to guide the baseline EMO algorithm. Note that this framework is so general that any existing EMO algorithm can be applied in a plug-in manner. Experiments on 48 benchmark test problems with up to ten objectives and a real-world multiobjective robot control problem fully demonstrate the effectiveness of our proposed algorithms for finding SOI.

Index Terms—Evolutionary multiobjective optimization (EMO), gradient descent, learning to rank (LTR), preference modeling.

I. INTRODUCTION

PARTIALLY due to the population-based property, evolutionary algorithms (EAs) have been widely recognized to be effective for multiobjective optimization problems (MOPs). Over the past three decades and beyond, many efforts have been dedicated to developing evolutionary multiobjective optimization (EMO) algorithms to obtain a set of tradeoff solutions that approximate a Pareto-optimal front (PF) with a decent diversity. Existing EMO algorithms can be classified into three categorizes, i.e., Pareto-, indicator-, and decomposition-based EMO approaches, where fast and elitist multiobjective genetic algorithm (NSGA-II) [1], indicator-based EA (IBEA) [2], and multiobjective EA based on decomposition (MOEA/D) [3] are representative algorithms, respectively. Approximating the entire PF can be a double-edged sword when it is handed over to the decision maker (DM) at the posteriori multicriterion decision-making (MCDM) stage. Due to the negligence of DM’s preference in the loop, it is not guaranteed to identify the solution(s) of interest (SOI) most relevant to the DM’s aspiration. This is further aggravated when the number of objectives becomes large given that the PF approximation is too sparse to cover the SOI, letting alone the cognitive barrier for understanding and interpreting the high-dimensional data.

Comparing to the posteriori decision making, a range of empirical studies reported in our recent paper [4] have shown the benefits of incorporating the DM’s preference information into the EMO process for locating the SOI. According to the preference elicitation manner, i.e., when to ask the DM to elicit/input her preference information, the existing preference-based EMO algorithms can be categorized into a priori EMO and interactive EMO. In view of the black box nature of real-world problems, of which the DM has little knowledge, it is controversial to elicit a reasonable preference information beforehand. Even worse, a disruptive preference information can lead to a failure of the underlying algorithm.

In contrast, the interactive EMO (e.g., [5], [6], [7], [8], and [9]) provides a better opportunity for the DM to progressively understand the underlying black box system, thus, to gradually amend her preference information. The DM is involved in the overall optimization-cum-decision-making loop as being periodically requested to input her preference information with respect to the selected solutions provided by the underlying EMO algorithm. Both direct (e.g., weights or an aspiration level vector [7]) and indirect information (e.g., score [6] or holistic pairwise comparison [8], [9], [10]) can be used to represent the DM’s preference. As discussed in [4], eliciting direct preference information is far from trivial when encountering a black box system. It is likely to be error prone and cognitively demanding when the number of objectives becomes large. In contrast, the indirect information have become more appealing and prevalent in the interactive EMO literature. Based on the inputs/feedback collected from the
DM, a preference model is progressively learned (e.g., radial basis function networks [6], nonlinear programming [10] and ordinal regression [8]), and it is used to guide the population toward the SOI.

Note that interactive multiobjective optimization approaches have been a longstanding area in the MCDM community [11] decades before the emergence of the EMO. In recent years, we have witnessed a growing trend of seeking synergies between EMO and MCDM. For example, our recent work [6] developed a framework for designing interactive EMO approaches via preference learning and it showed encouraging results on problems with up to ten objectives. Unfortunately, this work is merely valid for decomposition-based EMO approaches and it is yet applicable for the Pareto- and indicator-based EMO approaches. In fact, most, if not all, existing interactive EMO approaches are designed to be algorithm specific (e.g., [8] and [10] are designed for NSGA-II, [12] is designed for IBEA, [6] and [9] are designed for MOEA/D). The prevalent preference learning, from indirect information, in interactive EMO is mainly derived from mathematical programming and operations research approaches while machine learning approaches such as learning-to-rank (LTR) [13] has rarely been considered, except [14]. Note that LTR has been recognized as an effective tool to learn user preference in information retrieval and recommendation systems [15].

Built upon our previous work [6], this article develops a general framework to design interactive EMO algorithms that progressively learn the DM’s preference information from her feedback and adapt the learned preference to guide the population toward the SOI. As in [6], this framework consists of three modules, i.e., consultation, preference elicitation, and optimization.

1) As the interface by which the DM interacts with the EMO algorithm, the consultation module mainly aims to collect the DM’s preference information from her feedback and to build a preference model. In particular, this article considers an indirect preference information in the form of holistic pairwise comparisons of solutions. The preference model is built upon the collected comparison results, constituting the training data, by using an LTR neural network.

2) In view of the unique environmental selection mechanism of the underlying EMO algorithm, the preference model is usually not directly applicable. The preference elicitation module plays as the catalyst that translates the preference information learned in the consultation module into the form that can be used in the underlying EMO algorithm.

3) The optimization module can be any EMO algorithm in the literature. For proof-of-concept purposes, this article chooses three iconic algorithms from each of Pareto-, indicator-, and decomposition-based EMO approaches (i.e., NSGA-II, IBEA, and MOEA/D). The optimization module leverages the preference information learned from the preference elicitation module to search for the SOI. In the meanwhile, it periodically provides the consultation module with a set of selected candidates for preference learning.

4) To validate the effectiveness of our proposed framework, we instantiate three interactive EMO algorithms, denoted as I-NSGA-II/LTR, I-R2-IBEA/LTR, and I-MOEA/D/LTR. We compare their performance against four state-of-the-art (SOTA) interactive EMO algorithms, BC-EMO [14], NEMO-0 [8], I-MOEA/D-PLVF [6], and IEMO/D [9], on 48 benchmark problem instances with a range of different characteristics and a real-world multiobjective robot control problem.

For the remaining parts of this article, Section II provides some necessary background knowledge including a pragmatic overview of the related works of preference-based EMO in an interactive manner. Section III delineates the implementation detail of the proposed interactive EMO framework. Section IV introduces the experimental settings while the comparison results with respect to SOTA peer algorithms are presented and discussed in Section V. At the end, Section VI concludes this article and sheds some light on future directions.

II. PRELIMINARIES

This section starts with some basic definitions related to this article, followed by a pragmatic overview on some selected developments of preference-based EMO in an interactive manner. Interested readers are referred to [4], [16], and [17] for a more comprehensive survey.

A. Basic Definitions

The MOP considered in this article is formulated as follows:

\[
\begin{align*}
\min & \quad \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top \\
\text{subject to} & \quad \mathbf{x} \in \Omega
\end{align*}
\]

where \( \mathbf{x} = (x_1, \ldots, x_n)^\top \) is a \( n \)-dimensional decision vector and \( \mathbf{f}(\mathbf{x}) \) is an \( m \)-dimensional objective vector. \( \Omega \) is the feasible set in the decision space \( \mathbb{R}^n \) and \( \mathbf{F} : \Omega \to \mathbb{R}^m \) is the corresponding attainable set in the objective space \( \mathbb{R}^m \). Without considering the DM’s preference information, given two solutions \( \mathbf{x}^1, \mathbf{x}^2 \in \Omega \), \( \mathbf{x}^1 \) is said to dominate \( \mathbf{x}^2 \) if and only if \( f_i(\mathbf{x}^1) \leq f_i(\mathbf{x}^2) \) for all \( i \in \{1, \ldots, m\} \) and \( \mathbf{f}(\mathbf{x}^1) \neq \mathbf{F}(\mathbf{x}^2) \). A solution \( \mathbf{x} \in \Omega \) is said to be Pareto-optimal if and only if there is no solution \( \mathbf{x}^* \in \Omega \) that dominates it. The set of all Pareto-optimal solutions is called the Pareto-optimal set (PS) and their corresponding objective vectors constitute the PF. Accordingly, the ideal point is defined as \( \mathbf{z}^* = (z^*_1, \ldots, z^*_m)^\top \), where \( z^*_i = \min_{\mathbf{x} \in \Omega} f_i(\mathbf{x}) \), and the nadir point is defined as \( \mathbf{z}^{\text{nad}} = (z^{\text{nad}}_1, \ldots, z^{\text{nad}}_m)^\top \), where \( z^{\text{nad}}_i = \max_{\mathbf{x} \in \text{PS}} f_i(\mathbf{x}) \) \( \forall i \in \{1, \ldots, m\} \).

B. Related Works on Interactive EMO

The initial attempt to incorporate the DM’s preference into EMO can be traced back to the early 90s when Fonseca and Fleming [35] suggested to model the DM’s preference as a goal that indicates desirable levels of performance at each objective dimension. The early works on preference-based EMO mainly use a priori preference information where
the DM only “interact” with the algorithm once at the outset of evolution. Almost all a priori preference-based EMO approaches can be applied in an interactive manner. For example, the DM can periodically adjust the reference point to progressively guide the population toward the SOI. However, partially due to the use of direct preference information, such as reference point (also known as aspiration level vector)\[36\], \[37\], \[38\], \[39\], weights \[40\], \[41\], \[42\], and desirability function \[43\], these approaches are cognitively demanding and highly likely to be error prone when the DM is intensively involved in the optimization loop.

In contrast, the interactive EMO, which bridges the gap between two sibling communities, i.e., EMO and MCDM, endeavor to keep DMs in the optimization loop, thus, to enable a collaborative human–computer optimization paradigm. The key features of some important developments of interactive EMO are summarized in Table I. The following paragraphs elaborate upon them according to the types of preference information, all of which are in an indirect format.

1) Holistic Pairwise Comparisons: During the consultation stage, the DM is asked to input her preference over a pair of candidates $\langle x^a, x^b \rangle$ at a time, such as $x^a$ is better, worse, or indifferent over $x^b$. As a pioneer along this line, Phelps and Köksalan \[18\] proposed an interactive evolutionary meta-heuristic algorithm that translates the pairwise comparison results into a utility function as a weighted sum. In particular, the weights therein are estimated by solving a linear programming problem. Likewise, Deb et al.\[10\] developed an interactive NSGA-II that progressively learns an approximated value function by asking the DM to compare a set of solutions in a pairwise manner. Battiti and Passerini \[14\] proposed to use a high-order polynomial as the utility function whose parameters are estimated by a support vector machine. The approximated utility function is used to modify the Pareto dominance in the environmental selection of NSGA-II. Branke et al. \[8\] proposed to use a robust ordinal regression to learn a representative additive monotonic value function that is used to replace the crowding distance in NSGA-II. Later, the same authors proposed an improved version of NEMO-II that applies the Choquet integral as the DM’s preference model \[32\]. Recently, Tomczyk and Kadaźiński \[9\] proposed an interactive EMO based on MOEA/D. It employs the $L_\alpha$-norm as the preference model and uses a Monte Carlo simulation with a rejection sampling to generate a set of weight vectors compatible with the learned preference information.

2) Objective-Level Comparisons: This type of preference information mainly exploit the relationship and importance among different objectives contingent upon the DM’s preference. The first attempt along this line is \[19\] where Cvetković and Parmee developed a fuzzy preference relation that translates the pairwise comparisons among objectives into a weighted-dominance relation according to the relative importance of objectives. By comparing objectives, Jin and Sendhoff \[20\] proposed to use a fuzzy logic to convert the DM’s preference information into weight intervals. Shen et al. \[21\] proposed an interactive EMO algorithm that periodically asks the DM to specify the relative importance between pairs of objectives via linguistic terms. Thereafter, the collected preference information is used to construct a new fitness function derived from a fuzzy inference system. By asking the DM to classify objectives into up to five classes, Miettinen and Mäkelä \[5\] developed an interactive multiobjective optimization system dubbed WWW-NIMBUS to progressively search for the SOI. Later, Miettinen et al. \[22\] proposed the NAUTILUS method that starts the search process from the estimated nadir point and interactively improve all objectives. Note that the DM is able to control the interaction frequency and improvement rates at different objectives. Sindhya et al. \[23\] proposed to use an EA as the search engine in the NAUTILUS. Recently, Guo et al. \[24\] proposed an interactive EMO algorithm that uses a three-step process, called partitioning–updating–tracking, to search for the SOI. In particular, the quality of a solution is measured according to the satisfaction of the semantic-based relative importance of different objectives.

3) Tradeoff Specification: Tradeoff relation between different objective functions is another format to interpret the DM’s preference information. Yang et al. \[25\], \[26\], \[27\] proposed a series of methods to search for the SOI by taking the indifference tradeoffs elicited by the DM as the preference model. They proposed the GRIST method to estimate the gradient
of the utility function and project the gradient onto the tangent hyperplane of the PF. To promote the GRIST method for solving problems without nice mathematical properties, such as convexity and differentiability, Chen et al. [28] proposed to use an EA to replace the gradient descent method.

4) Polarized Solution(s) Selection: Asking the DM to periodically select the most preferred and/or the most dislike solution(s) from a set of candidates is another alternative way to represent the DM’s preference. For example, Folwer et al. [29] proposed a cone dominance relation based on convex preference cones by asking the DM to specify the best and the worst solutions from the current population. Sinha et al. [30] proposed a modified Pareto dominance relation based on polyhedral cones. It is built by asking the DM to select the most preferred solution(s) from an archive. By considering the uncertainty associated with the interactive EMO, Gong et al. [31] proposed to convert the uncertainty into interval parameters and developed a preference polyhedron, constructed by convex cones, to approximate the DM’s preference. Köksalan and Karahan [44] proposed an interactive version of territory defining EA [33] where a territory is defined around each solution and the favorable weights of the best solution selected by the DM are identified to determine a new preferred weight region. Gong et al. [34] proposed an interactive MOEA/D that uses the current best solution as an anchor to update the distribution of weight vectors to the ROI.

5) Performance Score: Another natural way to express the preference is scoring within a given range of numeric numbers. Recently, the first author and his collaborators [6] proposed an interactive EMO framework specifically designed for the decomposition multiobjective optimization. It periodically invites the DM to assign scores over some selected solutions according to their satisfaction to the DM’s preference. Based on the collected scoring results, a radial basis function network is applied to build the preference model and it is used as the fitness function to guide the population toward the SOI in the next several iterations. In particular, the preferred search directions are expressed as a set of weight vectors biased toward the potential SOI.

Remark 1: According to the above brief literature review and the summary in Table I, we can see that existing interactive EMO approaches are algorithm-specific. In other words, they are specifically designed upon a baseline algorithm, e.g., NSGA-II, IBEA, or MOEA/D. There is no solution applicable for all Pareto-, indicator-, and decomposition-based EMO frameworks. Note that our recent study [4] has shown that all EMO frameworks are well-suitable as a baseline for designing effective preference-based EMO algorithms.

Remark 2: Most, if not all, works use utility function as the preference model, the fitting of which is implemented as either a mathematical programming problem, a regression analysis, or a fuzzy logic. As a subfield in machine learning, LTR, as known as machine-learned ranking, is a powerful tool for preference learning from indirect information. It has been widely studied in information retrieval [13] and recommendation systems [15]. However, it has rarely been considered in the context of interactive EMO, except [14] to the best of our knowledge, which applied a support vector machine to serve the preference learning purpose.

Remark 3: Note that, before the development of interactive EMO, there have been a plethora of studies on interactive EAs that optimize systems based on subjective human evaluation [45]. Since they are mainly about single-objective optimization, of which the fitness function is determined without trading off conflicting objective functions, they are not directly applicable and out of the context of this article.

III. PROPOSED METHOD

The workflow of our proposed interactive EMO framework based on an LTR neural network is given in Fig. 1. Conceptually, this framework is a closed loop of three modules while its termination is either called out by the DM or the exhaustion of the computational budget. In the following paragraphs, we will delineate the implementation of each module step by step.

A. Consultation Module

The consultation module is the interface where the DM interacts with the EMO algorithm. The DM is asked to specify her preference over a set of selected candidate solutions \( S = \{\tilde{x}^i\}_{i=1}^\mu, 1 \leq \mu \ll N \). Then, a preference model is learned based on the collected preference information. To this end, we need to address the following three core questions.

1) Which Solutions Are Chosen for Consultation: As discussed in [6], it is arguable to simply ask the DM to compare all solutions in a population every generation. This makes the search be completely driven by the DM. Thus, it significantly increases her cognitive load and is highly likely to lead to fatigue. As discussed in [14], the DM can hardly make reasonable judgments on poorly converged solutions, thus, it might not be helpful or even detrimental to consult the DM at the early stage of evolution. In this article, we fix the number of consultations, say every \( \tau > 1 \) generations, after running an EMO algorithm without considering any DM’s preference information for several generations. During each consultation session, only a limited number of \( \mu \) incumbent solutions, evaluated by the utility function learned by our preference model introduced in Section III-A3, are chosen to constitute \( S \) for preference elicitation.

2) What Preference Information Do We Ask the DM To Give: As overviewed in Section II-B, there are multiple ways for the DM to specify her preference information. In this article, we consider the holistic pairwise comparisons as the indirect preference information. Specifically, during the consultation stage, the DM is asked to iteratively decide, according to her preference, the quality of a solution pair \( \langle \tilde{x}^i, \tilde{x}^j \rangle \) chosen from \( S \) where \( i, j \in \{1, \ldots, \mu \} \) and \( i \neq j \). The outcome of each pairwise comparison is either \( \tilde{x}^i \) is better, worse or
indifferent over \( \tilde{x}' \), denoted as \( \tilde{x}' > \tilde{x}' \), \( \tilde{x}' < \tilde{x}' \), or \( \tilde{x}' \simeq \tilde{x}' \). In total, there are \( \binom{n}{2} \) pairwise comparisons, thus, leading to \( \binom{n}{2} \) holistic indirect judgments.

3) **How To Learn Preference Model:** Based on the collected preference information, i.e., the holistic indirect judgments, the goal of preference learning is to learn a preference model that is able to evaluate the quality of solutions according to the DM’s preference. In principle, this model is a utility function \( u(x) : \mathbb{R}^m \rightarrow \mathbb{R} \) such that the ranking order of a set of testing samples satisfies that \( u(x') > u(x) \) if \( x' > x \) and \( u(x') = u(x) \) if \( x' \simeq x \).

LTR is a machine learning approach, typically as a supervised or a semi-supervised learning, to construct a ranking model. LTR has been a core part of modern information retrieval systems, such as document retrieval [46], collaborative filtering [47], sentiment analysis [48], and online advertising [49]. As the diagram shown in Fig. 2, the training data of an LTR model consist of lists of items with some partial orders specified by the user queries between items in each list. This order is typically induced by giving a numerical or ordinal score or a binary judgment, such as “relevant” versus “irrelevant” for each item. After learning the latent information from the training data, the LTR model comes up to constitute a ranking system that predicts the ranking order of a permutation of items in the new and unseen testing dataset.

According to the above description, we appreciate that LTR shares a common philosophy as our preference learning purpose. In this article, we propose to use LTR as an alternative preference model. The quality of the underlying preference model is measured by the relative comparison results between pairs of candidate solutions. In other words, the less inaccurate order of solution pairs found, the better preference model is. As the diagram shown in Fig. 2, the training order of solution pairs found, the better preference model is. In other words, the less inaccurate of items in the new and unseen testing dataset.

![Flowchart of a classic LTR routine.](image)

### Equations

1. **Cross-entropy between**
   
   \[
   \ell_{ij} = -\log \left( \frac{e^{\sigma (u(x') - u(x))}}{e^{\sigma (u(x) - u(x'))} + e^{\sigma (u(x') - u(x))}} \right)
   \]

2. **If** \( u(x') \) **equals** \( u(x) \) **but** \( x' \simeq x \) **does not conform**, we still have \( \ell_{ij} > 0 \) that penalizes this pair \( \langle x', x' \rangle \), thus, leading to their ranking apart from each other.

3. **The overall loss function is the summation of the loss function of each ranking pair**
   
   \[
   \mathcal{L} = \sum_{\langle x', x' \rangle \in I} \ell_{ij}
   \]

where \( I \) is the set of all ranking pairs and this loss function asymptotes to a linear function.

In this article, we apply a neural network with a single hidden layer to learn the preference model. Since \( \mathcal{L} \) is differentiable, stochastic gradient descent (SGD) [51] is used to update the weights \( \bar{w} = (\bar{w}_1, \ldots, \bar{w}_\ell)^T \) of the neural network

\[
\bar{w}_k = \bar{w}_k - \eta \frac{\partial \mathcal{L}}{\partial \bar{w}_k}
\]

**Note** that there are two characteristics of this loss function.

1. **If** \( u(x') \) **equals** \( u(x) \) **but** \( x' \simeq x \) **does not conform**, we still have \( \ell_{ij} > 0 \) that penalizes this pair \( \langle x', x' \rangle \), thus, leading to their ranking apart from each other.

2. **The overall loss function is the summation of the loss function of each ranking pair**

\[
\mathcal{L} = \sum_{\langle x', x' \rangle \in I} \ell_{ij}
\]

where \( I \) is the set of all ranking pairs and this loss function asymptotes to a linear function.

In this article, we apply a neural network with a single hidden layer to learn the preference model. Since \( \mathcal{L} \) is differentiable, stochastic gradient descent (SGD) [51] is used to update the weights \( \bar{w} = (\bar{w}_1, \ldots, \bar{w}_\ell)^T \) of the neural network

\[
\bar{w}_k = \bar{w}_k - \eta \frac{\partial \mathcal{L}}{\partial \bar{w}_k}
\]

thereby reducing \( \mathcal{L} \). If we further decompose the gradient term as follows:

\[
\frac{\partial \mathcal{L}}{\partial \bar{w}_k} = \sum_{\langle x', x' \rangle \in I} \left( \frac{\partial \mathcal{L}_{ij}}{\partial u(x')} \frac{\partial u(x')}{\partial \bar{w}_k} + \frac{\partial \mathcal{L}_{ij}}{\partial u(x')} \frac{\partial u(x')}{\partial \bar{w}_k} \right)
\]

where

\[
\frac{\partial \mathcal{L}_{ij}}{\partial u(x')} = \sigma \left( \frac{1}{2} (1 - c_{ij}) - \frac{1}{1 + e^{\sigma (u(x') - u(x))}} \right)
\]

\[
= -\frac{\partial \mathcal{L}_{ij}}{\partial u(x')}
\]

Let \( \lambda_{ij} = (\partial \mathcal{L}_{ij}/[\partial u(x')]) \), then we can rewrite (10) as follows:

\[
\frac{\partial \mathcal{L}}{\partial \bar{w}_k} = \sum_{\langle x', x' \rangle \in I} \lambda_{ij} \left( \frac{\partial u(x')}{\partial \bar{w}_k} - \frac{\partial u(x)}{\partial \bar{w}_k} \right)
\]

This formulation makes the training of our neural network as a mini-batch learning process which reduces the computational complexity of SGD from quadratic to linear.
4) Working Example: Here, we use a simple example to illustrate the working mechanism of our preference model learning process. For simplicity, let us assume that the unknown utility function is a weighted aggregation of different objectives, i.e., \( u(x) = \sum_{i=1}^{m} w_i f_i(x) \) where \( w_i > 0 \). We use the 3-objective DTLZ2 problem instance \([52]\) as an example and the middle of its PF is assumed to be preferred by the DM. Given four solutions \( \{x^i\}_{i=1}^{4} \) as listed as follows:

| \(x^i\) | \(f_1(x)\) | \(f_2(x)\) | \(f_3(x)\) | \(r(x)\) |
|---|---|---|---|---|
| \(x^1\) | 0.686 | 0.514 | 0.514 | 1 |
| \(x^2\) | 0.514 | 0.686 | 0.514 | 2 |
| \(x^3\) | 0.302 | 0.905 | 0.302 | 3 |
| \(x^4\) | 0.0 | 0.0 | 1.0 | 4 |

where \( r(x) = \{1, \ldots, 4\} \) is the preference rank of a solution. In particular, the smaller the \( r(x) \) is, the more preferred \( x \) is.

In particular, the smaller the \( \partial f_i \) is, the more preferred \( x \) is.

Let us set the initial weight as \( w = (0.1, 0.1, 0.1) \) and the learning rate of SGD is set as \( \eta = 0.001 \). After ten iterations, we have \( w^* = (0.11617, 0.10195, 0.09581) \) and the corresponding utility function values predicted by our preference model are listed as follows:

| \(x\) | \(\partial f_1\) | \(\partial f_2\) | \(\partial f_3\) | \(u^0(x)\) | \(u^{10}(x)\) |
|---|---|---|---|---|---|
| \(x^1\) | 0.172 | -0.172 | 0.0 | 0.1714 | 0.1813 |
| \(x^2\) | 0.384 | -0.391 | 0.212 | 0.1714 | 0.1789 |
| \(x^3\) | 0.514 | 0.686 | 0.486 |
| \(x^4\) | 0.0 | 0.0 | 1.0 | 0.1 | 0.0958 |

From the above-predicted utility function values, we can see that the preference order predicted by our preference model conforms to the ground truth.

5) Working Example: Here, we use a simple example to illustrate the learning process of our preference model. For simplicity, let us assume that the unknown utility function standing for the DM’s preference is the Tchebycheff function

\[
g(x|w, z^*) = \max_{1 \leq i \leq m} \frac{|f_i(x) - z^*_i|}{w_i}
\]

where we assume that \( z^* = 0 \) and \( w_i > 0 \). We use the 3-objective DTLZ1 \([52]\) as an example and the middle of its PF is assumed to be preferred by the DM, with \( w = (1, 1, 1)^T \).

Given four solutions \( \{x^i\}_{i=1}^{4} \) as listed as follows:

| \(x\) | \(f_1(x)\) | \(f_2(x)\) | \(f_3(x)\) | \(g(x|w)\) | \(r(x)\) |
|---|---|---|---|---|---|
| \(x^1\) | 0.167 | 0.167 | 0.167 | 0.167 | 1 |
| \(x^2\) | 0.2 | 0.15 | 0.15 | 0.2 | 2 |
| \(x^3\) | 0.3 | 0.1 | 0.1 | 0.3 | 3 |
| \(x^4\) | 0.4 | 0.05 | 0.05 | 0.4 | 4 |

where \( r(x) = \{1, \ldots, 4\} \) is the preference rank of a solution. In particular, the smaller the \( r(x) \) is, the more preferred \( x \) is.

We can have \( \binom{4}{2} = 12 \) pairwise comparisons in total while we only choose three pairs, i.e., \( x^1 > x^2, x^1 > x^3, \) and \( x^2 > x^4 \) to train our neural network.

In this case, the output of our preference model is a weighted aggregation as

\[
u(x) = \sum_{i=1}^{m} \tilde{w}_i f_i(x)
\]

where \( \tilde{w}_i \) is a weight that needs to be learned in our neural network training process. For example, let us assume that the initial weights are set as \( \tilde{w} = (0.620, -0.952, -0.734)^T \). After the training process, we have \( \tilde{w}^* = (-2.846, 2.524, 2.742)^T \), and the corresponding utility function values predicted by our preference model are listed as follows:

| \(x\) | \(f_1(x)\) | \(f_2(x)\) | \(f_3(x)\) | \(u^0(x)\) | \(u^*(x)\) |
|---|---|---|---|---|---|
| \(x^1\) | 0.167 | 0.167 | 0.167 | -0.178 | 0.404 |
| \(x^2\) | 0.2 | 0.15 | 0.15 | -0.129 | 0.221 |
| \(x^3\) | 0.3 | 0.1 | 0.1 | 0.017 | -0.327 |
| \(x^4\) | 0.4 | 0.05 | 0.05 | 0.164 | -0.875 |

where \( u^0(x) \) and \( u^*(x) \), respectively, indicate the preference scores assigned to solutions by the preference model before and after the neural network training. Note that the ground truth utility function values of \( \{x_i\}_{i=1}^{4} \) are unknown a priori when training the preference model. From this example, we can see that the rank inferred from the predicted preference scores conforms to the ground truth.

B. Preference Elicitation Module

The preference elicitation module transforms the preference information learned in the consultation module into the format that can be used in the optimization module. There are three major EMO frameworks, i.e., Pareto-, indicator-, and decomposition-based, in the literature. Each one differs from the others mainly in the environmental selection, i.e., the way of survival the fittest. In this section, we choose three iconic EMO algorithms from these three frameworks as the baseline and tailor our preference model to their environmental selection.

1) Pareto-Based EMO Algorithm: It usually consists of two parts: one is the use of Pareto dominance to push the population toward the PF (i.e., convergence) and the other is the use of a density estimation metric to maintain the
population diversity. Here, we choose NSGA-II, which has been widely recognized as one of the most successful algorithms, as the baseline for the Pareto-based EMO algorithm. In practice, we keep the fast nondominated sorting untouched since the Pareto optimality is always the first priority in multiobjective optimization. The utility function learned by our preference model in the consultation module is used as the alternative of crowding distance in NSGA-II.

2) Decomposition-Based EMO Algorithm: Here, we focus on MOEA/D which has been widely recognized as the iconic decomposition-based EMO algorithm. Its basic idea is to decompose the original MOP into a set of subproblems, either as scalarizing functions or simplified MOPs. Thereafter, a population-based meta-heuristic is applied to solve these subproblems in a collaborative manner. In MOEA/D, the weight vectors used to define the subproblems can be regarded as the driver to represent the DM’s preference information. In particular, each weight vector indicates a preferred region on the PF. The preference elicitation module mainly aims to change the originally uniformly distributed weight vectors \( W = \{w_i\}_{i=1}^N \) to be biased toward the region of interest. Here, we follow the same four-step procedure developed in our recent work [6] to serve this purpose.

Step 1: Use \( u(x) \) learned in the consultation module to score each member of the current population \( P \).

Step 2: Rank the population according to the scores assigned in step 1, and find the top \( \mu \) solutions. Weight vectors associated with these solutions are deemed as the promising ones, and store them in a temporary archive \( W^U = \{w^U_i\}_{i=1}^\mu \) where \( 1 \leq \mu' \leq \mu \).

Step 3: For \( i = 1 \) to \( \mu' \) do

- Step 3.1: Find the \( \lceil (N - \mu'/\mu') \rceil \) closest weight vectors to \( w^U_i \) according to their Euclidean distances.

- Step 3.2: Move each of these weight vectors toward \( w^U_i \) as follows:

\[
w_j = w_j + \eta \times (w^U_j - w_j) \tag{14}
\]

where \( j \in \{1, \ldots, m\} \).

- Step 3.3: Temporarily remove these weight vectors from \( W \) and go to Step 3.

Step 4: Output the adjusted reference points as the new \( W \).

In particular, \( 0 < \eta \leq 1 \) is the step size used to tweak the weight vectors. Fig. 4 gives an example of this preference elicitation process in a two-objective case. Three promising reference points are highlighted by red circles. \( w^U \) has the highest priority to attract its companions, and so on. Interested readers are referred to [6] for more detail.

Remark 4: From our preliminary experiments, we find that the distribution of the initial weight vectors has a significant impact on the weight vector adjustment in step 3. This can be attributed to the sparse distribution of weight vectors generated by the widely used Das and Dennis’ method [53]. As the 8- and 10-objective examples shown in Fig. 5, we can see that almost all weight vectors are sparsely distributed along the boundary of the simplex while there are few lying in the middle part. This renders the weight adjustment in step 3 hardly effective in a high-dimensional scenario. To mitigate this issue, inspired by [9], we apply the hit-and-run (HAR) method [54] as an alternative of Das and Dennis’ method for the initialization of weight vectors. Note that the HAR method is a variant of a Markov Chain Monte Carlo and is scalable to a high-dimensional space [55]. Its basic idea is to sample as a set of uniformly distributed weight vectors from a constrained simplex. Specifically, for each weight vector \( w^i \in W \), \( i \in \{1, \ldots, N\} \), it should satisfy some linear constraints \( \sum_{i=1}^{m} w^i_j = 1 \) and \( w^l_j \leq w^i_j \leq w^u_j \) where \( w^l \) and \( w^u \) are the lower and upper bounds for the \( j \)th component of the weight vector \( w^i \). In particular, we set \( w^l = 0 \) and \( w^u = 1 \) for \( j \in \{1, \ldots, m\} \) in this article. From the 8- and 10-objective examples shown in Fig. 5, we can see that weight vectors generated by the HAR method have a descent distribution in the intermediate section of the simplex.

3) Indicator-Based EMO Algorithm: Its basic idea is to apply a performance indicator to transform an MOP into a single-objective optimization problem. Instead of using the binary \( \epsilon \)-indicator as in the classic IBEA, we opt to the \( R_2 \) indicator [56] in view of its encouraging results reported for multi- and many-objective optimization [57] as well as preference articulation [58].

Specifically, \( R_2 \) indicator is used to evaluate the relative quality of two sets of tradeoff solutions by using the weighted Tchebycheff function with a given reference point \( z = (z_1, \ldots, z_m)^T \).
where $\mathcal{P}$ is the current population, $\mathcal{W}$ is a set of weight vectors, and $\mathbb{P}(\mathcal{W})$ indicates the probability distribution on $\mathcal{W}$. In particular, the $R_2$ indicator can be rewritten as follows when the weight vectors are evenly distributed:

$$
R_2(\mathcal{P}, \mathcal{W}, z) = \sum_{i=1}^{m} \left( \mathbb{P}(\mathcal{W}) \times \min_{x^* \in \mathcal{P}} \left\{ \max_{1 \leq j \leq m} w^j |x^*_j - z_j| \right\} \right)
$$

In this case, we can see that the DM’s preference information can also be represented as a set of biased weight vectors. In other words, we use a set of adjusted weight vectors obtained by the four-step procedure introduced in Section III-B2 to replace the $\mathcal{W}$ in (16).

### C. Optimization Module

In principle, any EMO algorithm can be adapted to be a baseline algorithm in the optimization module. This article chooses NSGA-II, MOEA/D, and R2-IBEA, as discussed in Section III-B for proof-of-concept purposes, to generate three algorithm instances of our proposed framework, denoted as I-NSGA-II/LTR, I-MOEA/D/LTR, and I-R2-IBEA/LTR, respectively. Note that all these algorithm instances run as the vanilla version without considering the DM’s preference before the first consultation session.

### IV. EXPERIMENTAL SETTINGS

This section gives the experimental settings of our empirical studies, including benchmark problems, parameter settings, peer algorithms, performance metrics, and statistical tests.

#### A. Benchmark Problem Suite

In this article, we consider test problems chosen from five widely used benchmark suites, including DTLZ1 to DTLZ6 [52], DTLZ1$^{-1}$ to DTLZ4$^{-1}$ [59], mDTLZ1 to mDTLZ4 [60], and WFG3 [61]. All these test problems are with continuous variables and have various PF shapes (e.g., linear, convex, concave, disconnected, degenerate, and inverted PFs) and different search space properties. In our experiments, we consider $m \in \{3, 5, 8, 10\}$ except the mDTLZ problems which are constantly with three objectives. The number of variables is set as recommended in their original papers.

#### B. Parameter Settings

The parameters associated with our proposed interactive EMO algorithms are outlined as follows.

1) The number of incumbent candidates presented to the DM for pairwise comparisons: $\mu = 10$.
2) The number of generations between two consecutive consultation sessions: $\tau = 10$.
3) The step size of the reference point update used in (14): $\eta = 0.2$.
4) The number of function evaluations (FEs) and population size settings are given in Table IV of the supplemental material\(^1\) as suggested in [9].
5) The crossover probability and the distribution index for the simulated binary crossover operator [62]: $p_c = 1.0$ and $\eta_c = 30$.
6) The mutation probability and the distribution index for the polynomial mutation operator [63]: $p_c = 1/n$ and $\eta_m = 20$.
7) The control parameter of the sigmoid function: $\sigma = 1$.

### C. Peer Algorithms

To validate the competitiveness of the proposed interactive EMO algorithms, we compare their performance with four SOTA peer algorithms in the literature, i.e., BC-EMO [14], NEMO-0 [8], I-MOEA/D-PLVF [6], and IEMO/D [9]. Note that BC-EMO is the only peer algorithm, to the best of our knowledge, that applies LTR to serve the preference learning purpose. Although the representation of the DM’s preference information in I-MOEA/D-PLVF is different from this article, it is still worthwhile to be compared since it is the first instantiation of the interactive EMO framework shown in Fig. 1. The corresponding parameters are set according to the recommendations in their original papers. Interested readers are referred to the corresponding papers for more technical details of these peer algorithms.

#### D. Performance Evaluation

1) Performance Metrics: As discussed in [64], performance evaluation of interactive EMO methods is far from trivial as the choice of DM model can lead to potential bias. This article first considers a prescribed golden value function, unknown to an interactive EMO algorithm, to play as the artificial DM

$$
\psi(x) = \max_{1 \leq i \leq m} \left| f_i(x) - z^*_i \right| / w^*_i
$$

where $z^* = (z_1, \ldots, z_m)^T$ is set to be the origin in our experiments, and $w^* = (w_1^*, \ldots, w_m^*)^T$ is the utopia weight that represents the DM’s expected importance of different objectives. In this article, we consider two types of $w^*$: one prefers the solution with an equal importance priority over all objectives (denoted as $w^e$) while the other one prefers the solution with a focused priority over a particular objective (denoted as $w^b$), i.e., biased toward a particular side of the PF. Since a $m$-objective problem has $m$ sides, there can be $m$ different choices for setting the biased $w^b$. In our experiments, we randomly choose one side for proof-of-concept purposes.

To evaluate the performance of an interactive EMO algorithm for approximating the ROI, we consider using the approximation error of the obtained population $\mathcal{P}$ with respect to the DM’s golden point $x^*$ (i.e., the Pareto-optimal solution for a given utopia weight) as the performance metric

$$
E(\mathcal{P}) = \min_{x^*} \text{dist}(x, x^*)
$$

where dist$(x, x^*)$ is the Euclidean distance between $x^*$ and a solution $x \in \mathcal{P}$ in the objective space. The choice of $w^e$ and $x^*$ are listed in Tables V to VIII of the supplemental material but is unknown to the algorithm.

\(^1\)The supplemental materials can be found in https://tinyurl.com/25v7wma.
2) Statistical Tests: Each experiment is repeated independently 31 times with different random seeds. To have a statistical interpretation of the significance of comparison results, three statistical measures are used in our empirical study.

1) Wilcoxon Signed-Rank Test [65]: This is a nonparametric statistical test that makes a little assumption about the underlying distribution of the data and has been recommended in many empirical studies in the EA community [66]. In particular, the significance level is set to $p = 0.05$ in our experiments.

2) Scott-Knott Test [67]: Instead of merely comparing the raw $E(P)$ values, we apply the Scott-Knott test to rank the performance of different peer techniques over 31 runs on each experiment. In a nutshell, the Scott-Knott test uses a statistical test and effect size to divide the performance of peer algorithms into several clusters. In particular, the performance of peer algorithms within the same cluster is statistically equivalent. Note that the clustering process terminates until no split can be made. Finally, each cluster can be assigned a rank according to the mean $E(P)$ values achieved by the peer algorithms within the cluster. In particular, since a smaller $E(P)$ value is preferred, the smaller the rank is, the better performance of the technique achieves.

3) $A_{12}$ Effect Size [68]: To ensure the resulted differences are not generated from a trivial effect, we apply $A_{12}$ as the effect size measure to evaluate the probability that one algorithm is better than another. Specifically, given a pair of peer algorithms, $A_{12} = 0.5$ means they are equivalent. $A_{12} > 0.5$ denotes that one is better for more than 50% of the times. $0.56 \leq A_{12} < 0.64$ indicates a small effect size while $0.64 \leq A_{12} < 0.71$ and $A_{12} \geq 0.71$ mean a medium and a large effect size, respectively. Note that both Wilcoxon signed-rank test and $A_{12}$ effect size are also used in the Scott-Knott test for generating clusters.

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

We seek to answer the following research questions (RQs) through our empirical study in the following paragraphs.

1) $RQ1$: How are the performance comparisons among our proposed three algorithm instances?

2) $RQ2$: How is the performance of our proposed algorithm instances compared against the selected SOTA peers?

3) $RQ3$: How is the performance of our proposed LTR neural network compared against the LTR algorithms used in NEMO–0 and BC–EMO?

4) $RQ4$: What is the impact of the hyper-parameters associated with our proposed framework?

A. Performance Comparison Among Our Proposed Three Algorithm Instances

Let us first look into the Wilcoxon signed-rank test results of $E(P)$ shown in Tables IX to XXIII in the supplemental document. As shown in Table I, we find that I-NSGA-II/LTR and I-MOEA/D/LTR are the most competitive algorithms on DTLZ1 to DTLZ6. In particular, all three proposed algorithm instances are more capable of finding the SOI with respect to an equal preference (i.e., $w^a$) than a biased preference (i.e., $w^b$). As the sample results shown in Fig. 6, we can see that I-NSGA-II/LTR and I-R2-IBEA/LTR are struggling on DTLZ4, which is featured with an induced bias on certain objective functions. As the sample results shown in Fig. 7, we find that I-NSGA-II/LTR can hardly converge to the SOI when $m = 8$ and $m = 10$. DTLZ1$^{-1}$ to DTLZ4$^{-1}$ have an inverted PF shape as opposed to DTLZ1 to DTLZ4. From the results shown in Table II, we find that I-NSGA-II/LTR has shown much better performance than the other two peer algorithms in most cases even for DTLZ4$^{-1}$ which is also featured with an induced bias as DTLZ4 (Fig. 8 gives the sample results on DTLZ4$^{-1}$ with $m = 10$). mDTLZ1 to mDTLZ4 have the same PF shape as DTLZ1$^{-1}$ to DTLZ4$^{-1}$ but they are featured with hardly dominated boundary solutions. From the results shown in Tables XIX to XXII of the supplemental document, I-MOEA/D/LTR becomes a competitive peer algorithm in most cases. WFG3 is a challenging problem [61] whose PF is partially degenerated to a lower-dimensional manifold. As the results in Table XXIII of the supplemental document and Fig. 9, we find that I-NSGA-II/LTR and I-MOEA/D/LTR have shown a better performance comparing to I-R2-IBEA/LTR.

In addition to the pairwise comparison conducted by the above Wilcoxon signed-rank test, we apply the Scott-Knott test to facilitate a better ranking among three proposed algorithm instances. Due to the large number of test problem instances used in our experiments, it will be messy if we list all ranking results ($48 \times 2 \times 3 = 288$ in total) obtained by the Scott-Knott test collectively. Instead, to have a better interpretation of the comparison among our proposed three algorithm instances based on LTR, we pull all the Scott-Knott test results together and show their distribution and variance as the box plots in Fig. 10(a). From this result, we find that I-NSGA-II/LTR is the best algorithm instance to approximate the SOI as it has been classified into the best.
Fig. 7. Population distribution of nondominated solutions with the medium $E(P)$ value obtained by I-NSGA-II/LTR, I-R2-IBEA/LTR, and I-MOEA/D/LTR on DTLZ6 when $m = 8$ and $m = 10$, respectively.

Fig. 8. Population distribution of nondominated solutions with the medium $E(P)$ value obtained by I-NSGA-II/LTR, I-R2-IBEA/LTR, and I-MOEA/D/LTR on DTLZ4 when $m = 10$.

Fig. 9. Population distribution of nondominated solutions with the medium $E(P)$ value obtained by I-NSGA-II/LTR, I-R2-IBEA/LTR, and I-MOEA/D/LTR on WFG3 when $m = 10$.

group in most comparisons. In contrast, I-R2-IBEA/LTR is the worst algorithm instance. It is worth noting that both I-R2-IBEA/LTR and I-MOEA/D/LTR use the same preference elicitation method, i.e., a set of biased weight vectors representing the DM’s preference information with respect to the ROI. In this case, the inferior results of I-R2-IBEA/LTR might be attributed to the relatively poor selection pressure provided by the $R^2$ indicator.

As discussed above, I-NSGA-II/LTR stands out as the best algorithm instance of our proposed framework. To better understand the performance difference of I-NSGA-II/LTR with respect to I-R2-IBEA/LTR and I-MOEA/D/LTR, we investigate the comparison results of $A_{12}$ effect size between I-NSGA-II/LTR and the other two peer algorithms, respectively. From the bar charts shown in Fig. 10(b), we find that the better results achieved by I-NSGA-II/LTR against I-R2-IBEA/LTR and I-MOEA/D/LTR are more evident than I-MOEA/D/LTR given that more than half of the better results are classified to be statistically large. This also supports the observations from the Scott-Knott test.

Answers to RQ1: We have the following takeaways from our experiments. 1) Among our proposed three algorithm instances, I-NSGA-II/LTR is the most competitive one for approximating the SOI in most cases. This is surprising at the first glance since the baseline NSGA-II is notorious for its poor scalability in problems with more than three objectives. 2) Although I-R2-IBEA/LTR and I-MOEA/D/LTR share the same preference elicitation method, the performance of I-MOEA/D/LTR is superior to that of I-R2-IBEA/LTR. This suggests that the Tchebycheff function can provide a stronger selection pressure than the $R^2$ indicator. 3) The solutions approximated by I-R2-IBEA/LTR and I-MOEA/D/LTR are usually...
Algorithm Instances Against the Other Peer Algorithms

This section investigates the performance of our proposed three algorithm instances with respect to four peer algorithms. Let us again look into the Wilcoxon signed-rank test results of $\mathbb{E}(P)$ shown in Tables XXIV to XXXVIII in the supplemental document. Note that NEMO-0 only has results for up to 8 objectives. This can be partially attributed to the exponentially soaring complexity of the linear programming involved in NEMO-0 that renders it extremely slow when the number of objectives becomes larger. Comparison results show that all our three proposed algorithm instances, even for the least competitive one I-R2-IBEA/LTR, have shown superior performance against the peer algorithms in most cases.

Like in Section V-A, we also apply the Scott-Knott test to investigate the collected ranking relation ($48 \times 2 \times 7 - 11 \times 2 = 650$ in total) among our proposed algorithm instances and the peer algorithms. From the box plots shown in Fig. 11, we can have a better view to see that all our three proposed algorithm instances are ranked as the most competitive algorithms while I-NSGA-II/LTR is still the best one. As for the selected peer algorithms, the performance of NEMO-0 and IEMO/D are better than that of I-MOEA/D-PLVF and BC-EMOA.

Last but not the least, we pick up each of our proposed three algorithm instances as a sentinel, respectively, and compare its performance difference with the other four peer algorithms by using the $A_{12}$ effect size. From the collected comparison results ($48 \times 2 \times 3 \times 3 + 37 \times 2 \times 3 = 1086$ in total) shown in Fig. 12, we can see that all our three proposed algorithm instances have shown overwhelming advantages over BC-EMOA and I-MOEA/D-PLVF as the sum of the percentage of the effect size is always close to 90%. When comparing with IEMO/D, the advantages of our proposed algorithm instances have been narrowed down. This can be partially attributed to the competitive performance of IEMO/D when the number of objectives is small. As for NEMO-0, it has shown certain comparable performance with our proposed algorithm instances. In particular, it has achieved more superior results against I-R2-IBEA/LTR and I-MOEA/D/LTR. These are also largely because of the better performance achieved by NEMO-0 when the number of objectives is small.

As expected, the performance of interactive EMO algorithms deteriorate with the increase of the number of objectives. However, it is interesting to find that NSGA-II, which is notorious for MOPs with more than three objectives, becomes surprisingly more resilient than MOEA/D as a baseline algorithm. This can be partially attributed to the increasing difficulties to: 1) learn a preference model due to the curse of dimensionality and short of human-labeled data and 2) specify appropriate weight vectors in a high-dimensional space.

Answers to RQ2: We have the following takeaways from our experiments. 1) All our three proposed algorithm instances have outperformed the other four peer algorithms in most comparisons. 2) Some of the selected peer algorithms, IEMO/D and NEMO-0 in particular, have shown competitive performance when the number of objectives is small. 3) Comparing to MOEA/D, NSGA-II is a more resilient baseline algorithm to constitute an interactive EMO algorithm with the increase of the number of objectives.

Performance Comparison of the LTR Neural Network Against the Other Peer Ranking Algorithms

From the results discussed in Sections V-A and V-B, we confirm that I-NGSA-II/LTR is the most competitive algorithm for approximating the SOI. The core difference between I-NGSA-II/LTR and BC-EMOA and NEMO-0 (the selected peer algorithms in Section V-B who also use NSGA-II as
the baseline algorithm) is the model used to learn the ranking among candidate solutions. In particular, BC-EMO uses the ranking SVM while NEMO-0 uses the ordinal regression. To address RQ3, we first create a set of synthetic data \( S = \{1\}^N \). In particular, \( t = (t_1, \ldots, t_m) \) where \( t_j \) is uniformly sampled from [0, 1], \( j \in \{1, \ldots, m\} \) and we consider \( m \in \{2, 3, 5, 8, 10\} \), respectively, in our experiment. During the experiment, we randomly pick up 50 sample pairs from \( S \) to constitute the training dataset for each LTR algorithm. To have a quantitative comparison, we apply the normalized discounted cumulative gain (NDCG) metric [69], widely used in the information retrieval domain, to evaluate the LTR performance.

From the comparison results shown in Table II, it is clear to see that the ranking SVM is the worst ranking algorithm. This explains the inferior performance of BC-EMO as discussed in Section V-B. In contrast, the performance of ordinal regression and LTR neural network is comparable. This observation also confirms the competitive results of NEMO-0 for approximating the SOI as shown in Fig. 12. However, we argue that the ordinal regression in NEMO-0 is not flexible enough to represent the DM’s preference information. In Appendix A of the supplemental document, we present a counterexample that shows the inability of the ordinal regression to learn an appropriate additive value function even when the DM provides accurate and consistent preference information.

Answers to RQ3: We have the following takeaways from this experiment. 1) The performance of an LTR algorithm can have impacts to the effectiveness of the interactive EMO algorithm. 2) Ranking SVM is less capable than the LTR neural network and the ordinal regression. However, the latter is not robust to learn an appropriate additive value function, which finally renders its ineffectiveness.

D. Sensitivity Study of Parameters

There are three parameters associated with our proposed interactive EMO framework based on an LTR neural network, i.e., \( \mu \), \( \tau \), and \( \eta \) as introduced in Section IV-B. In particular, \( \mu \) and \( \tau \) are only related to I-NSGA-II/LTR while I-MOEA/D/LTR and I-R2-IBEA/LTR involve all three parameters. In this section, we plan to empirically investigate the sensitivity of the performance of our three algorithm instances with respect to these parameters. To this end, we consider different settings of \( \tau = \{5, 10, 20\} \), \( \mu = \{5, 10, 20\} \), and \( \eta = \{0.1, 0.2, 0.4\} \) while the other parameters are kept the same as introduced in Section IV-B and the experiments are conducted on the benchmark problems as introduced in Section IV-A.

1) Effect of \( \tau \): It controls the number of generations between two consecutive consultation sessions. Specifically, a small \( \tau \) means that we need to frequently consult the DM about pairwise comparisons. By doing so, we can in principle expect a more accurate preference model due to the increased amount of labeled data. However, from the box plots shown in Fig. 13, it is surprising to see that the overall performance of LTR neural network is significantly better than the other peers according to the Wilcoxon’s rank sum test at a 0.05 significance level; \( \dagger \) denotes the corresponding algorithm significantly outperforms the LTR neural network. NDCG@20 indicates that the DM is only interested in the correctness of the top-20 ranking results.

Fig. 13. Box plots of Scott-Knott test ranks achieved by different settings of \( \tau \), \( \mu \), and \( \eta \) (the smaller rank is, the better performance is achieved). The distribution of the obtained ranks is presented as the black ● symbol in the corresponding boxes. The median value is highlighted as a circle ○ in each box.
but may have a risk of premature convergence toward an undesired region. On the contrary, a small \( \eta \) may slow down the convergence toward the SOI within the limited number of FEs. It can be observed from the box plots in Fig. 13 that a smaller \( \eta \) leads to better results in most cases.

**Answers to RQ4:** We have the following takeaways from our experiments. 1) It is not recommended to consult the DM too frequently as this not only increases the risk of making the DM fatigue but also introduces extra noises to the search process. 2) It is anticipated to improve the accuracy of the LTR model by involving more labeled data from the DM. However, this brings extra queries that inevitably increase the DM’s workloads. 3) A small step size \( \eta \) can be beneficial to fine tune the search direction toward the SOI. However, this may also lead to a slow convergence toward the SOI.

**E. Further Experiments**

At the end, we further evaluate the following two additional characteristics of our proposed interactive EMO framework.

1) **Impact of Inconsistency in the Elicited Preference:** In practice, it is not uncommon that practical decision-making and preference elicitation can be largely inconsistent. In other words, there exist a certain level of noises to which the pairwise comparison results can be conflicting with respect to the ground truth. Here, we investigate such impact as the results shown in Fig. 14. We find that: 1) the performance of I-NSGA-II/LTR and I-R2-IBEA/LTR are resilient to the induced noise in the preference elicitation and 2) I-MOEA/D/LTR can be impaired by involving a large noise in the preference elicitation but it is still robust to some mild inconsistencies. Detailed results are discussed in Appendix B of the supplemental document.

2) **Case Study on Real-World Application:** At the end, we validate the performance of I-NSGA-II/LTR, the best algorithm instance reported in Section V-A against the other peer algorithms on a multiobjective robot control problem built upon the MuJoCo Swimmer environment.\(^2\) From the box plots of \( \mathbb{E}(P) \) shown in Fig. 15, we can see that our proposed I-NSGA-II/LTR is the best algorithm to guide the MORL toward the DM-preferred policies. It is interesting to note that NEMO-O is the second-best algorithm and is way better than the other three peer algorithms. This result supports on the observation in Section V-B and it also demonstrates that the ordinal regression is a competitive preference learning method as reported in Section V-C. Details can be found in Appendix C of the supplemental document.

**VI. CONCLUSION**

This article developed a general framework to design interactive EMO algorithms that progressively learn the DM’s preference information from her feedback and adapt the learned preference to guide the population toward the SOI. It consists of three modules, i.e., consultation, preference elicitation, and optimization. As an interface between the DM and the EMO algorithm, the consultation module collects the implicit preference information (in the form of pairwise comparisons), based on which it learns a latent preference model. Once the DM’s latent preference information is learned, the preference elicitation module translates it into a tailored form that can be used in any prevalent EMO algorithms according to their environmental selection mechanisms. Experiments on benchmark test problems and a real-world application fully demonstrated the effectiveness of our proposed framework for helping three iconic EMO algorithms for finding the DM’s preferred solution(s).

As discussed in [4], the synergy of ideas between EMO and MCDM is an exciting direction to push the boundary of multiobjective optimization and decision making. This article can be extended from at least three aspects. First, since the physical queries can be labor intensive and error prone, it is interesting to investigate active learning mechanisms[70] to pick up the most informative samples in a strategic manner. In view of the black-box nature of real-world optimization problems, the explainability have rarely been explored in the literature. It is worthwhile to investigate the interpretability

---

\(^2\)https://mujoco.readthedocs.io/en/latest/overview.html
of the obtained SOI along with the tradeoff among conflicting objectives [71]. This can facilitate the understanding of the DM’s latent preference information and further advance a better-informed MCDM. Last but not the least, advanced techniques developed in the human–computer interaction domain can be leveraged to realize a human–machine symbiosis in future [72].

REFERENCES

[1] K. Deb, A. Pratap, S. Agrawal, 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.

[2] E. Zitzler and S. Künzli, “Indicator-based selection in evolutionary multiobjective optimization using reference points,” IEEE Trans. Evol. Comput., vol. 24, no. 6, pp. 1078–1096, Dec. 2020.

[3] K. Miettinen and M. M. Mäkelä, “Interactive multiobjective optimization system WWW-NIMBUS on the Internet,” Comput. Oper. Res., vol. 27, no. 7–8, pp. 709–723, 2000.

[4] K. Li, R. Chen, D. A. Savić, and X. Yao, “Interactive decomposition multiobjective optimization via progressively learned value functions,” IEEE Trans. Fuzzy Syst., vol. 27, no. 5, pp. 849–860, May 2019.

[5] K. Deb and A. Kumar, “Light beam search based multi-objective optimization using evolutionary algorithms,” in Proc. IEEE Conge. Evol. Comput. (CEC), 2007, pp. 2125–2132.

[6] J. Branke, S. Greco, R. Słowiński, and P. Zielniewicz, “Learning value functions in interactive evolutionary multiobjective optimization,” IEEE Trans. Evol. Comput., vol. 24, no. 2, pp. 320–334, Apr. 2020.

[7] K. Deb, A. Sinha, P. J. Korhonen, and J. Wallenius, “An interactive evolutionary multiobjective optimization method based on progressively approximated value functions,” IEEE Trans. Evol. Comput., vol. 14, no. 5, pp. 723–739, Oct. 2010.

[8] K. Miettinen, Nonlinear Multiobjective Optimization, vol. 12. Dordrecht, The Netherlands: Kluwer Acad. Publ., 1999.

[9] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, “An interactive simple indicator-based evolutionary algorithm (i-SBREA) for multiobjective optimization problems,” in Proc. 8th Int. Conf. Evol. Multi-Criterion Optim. (EMO), 2015, pp. 277–291.

[10] T.-Y. Liu, “Learning to rank for information retrieval,” Found. Trends Inf. Retrieval, vol. 3, no. 3, pp. 225–331, 2009.

[11] R. Battiti and A. Passerini, “Brain-computer evolutionary multiobjective optimization: A genetic algorithm adapting to the decision maker,” IEEE Trans. Evol. Comput., vol. 14, no. 5, pp. 671–687, Oct. 2010.

[12] M. de Gemmis, L. Iaquinta, P. Lops, C. Musto, F. Narducci, and G. Semeraro, “Learning preference models in recommender systems,” in Preference Learning. Berlin, Germany: Springer, 2010, pp. 387–407.

[13] B. Xin, L. Chen, J. Chen, H. Ishibuchi, K. Hirota, and B. Liu, “Interactive multiobjective optimization: A review of the state-of-the-art,” IEEE Access, vol. 6, pp. 41256–41279, 2018.

[14] S. P. Phelps and M. Köksalan, “An interactive evolutionary metaheuristic for multiobjective combinatorial optimization,” Manag. Sci., vol. 51, no. 12, pp. 1726–1738, 2005.

[15] D. Cvetković and I. C. Parmee, “Preferences and their application in evolutionary multiobjective optimization,” IEEE Trans. Evol. Comput., vol. 6, no. 1, pp. 42–57, Feb. 2002.

[16] Y. Jin and B. Sendhoff, “Incorporation of fuzzy preferences into evolutionary multiobjective optimization,” in Proc. Genet. Evol. Comput. Conf. (GECCO), 2002, p. 683.

[17] X. Shen, Y. Guo, Q. Chen, and W. Hu, “A multi-objective optimization algorithm incorporating preference information based on fuzzy logic,” Comput. Opt. Appl., vol. 46, no. 1, pp. 159–188, 2010.

[18] K. Miettinen, P. Eskelinen, F. Ruiz, and M. Luque, “NAUTILUS method: An interactive technique in multiobjective optimization based on the nadir point,” Eur. J. Oper. Res., vol. 206, no. 2, pp. 426–434, 2010.

[19] K. Sindhya, A. B. Ruiz, and K. Miettinen, “A preference based interactive evolutionary algorithm for multi-objective optimization: PIE,” in Proc. 6th Int. Conf. Evol. Multi-Criterion Optim. (EMO), 2011, pp. 212–225.

[20] Y. Guo, X. Zhang, D.-W. Gong, Z. Zhang, and J.-J. Yang, “Novel interactive preference-based multiobjective evolutionary optimization for bolt supporting networks,” IEEE Trans. Evol. Comput., vol. 24, no. 4, pp. 750–764, Aug. 2020.

[21] J.-B. Yang, “Gradient projection and local region search for multiobjective optimization,” Eur. J. Oper. Res., vol. 112, no. 2, pp. 432–459, 1999.

[22] J.-B. Yang and D. Li, “Normal vector identification and interactive trade-off analysis using minimax formulation in multiobjective optimization,” IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 32, no. 3, pp. 305–319, May 2002.

[23] M. Luque, J. Yang, and B. Y. H. Wong, “PROJECT method for multiobjective optimization based on gradient projection and reference points,” IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 39, no. 4, pp. 864–879, Jul. 2009.

[24] L. Chen, B. Xin, and J. Chen, “A trade-off based multi-objective optimization method driven by evolutionary algorithms,” J. Adv. Comput. Intell. Inform., vol. 21, no. 2, pp. 284–292, 2017.

[25] J. W. Fowler, E. S. Gel, M. Köksalan, P. J. Korhonen, J. L. Marquis, and J. Wallenius, “Interactive evolutionary multi-objective optimization for quasi-concave preference functions,” Eur. J. Oper. Res., vol. 206, no. 2, pp. 417–425, 2010.

[26] A. Sinha, P. J. Korhonen, J. Wallenius, and K. Deb, “An interactive evolutionary multi-objective optimization method based on polyhedral cones,” in Proc. 4th Int. Conf. Learn. Intell. Optim. (LION), 2010, pp. 318–332.

[27] D. Gong, J. Sun, and X. Ji, “Evolutionary algorithms with preference polyhedron for interval multi-objective optimization problems,” Inf. Sci., vol. 233, pp. 141–161, Jun. 2013.

[28] J. Branke, S. Corrente, S. Greco, R. Słowiński, and P. Zielniewicz, “Using Choquet integral as preference model in interactive evolutionary multi-objective optimization,” Eur. J. Oper. Res., vol. 250, no. 3, pp. 884–901, 2016.

[29] I. Karahan and M. Köksalan, “A territory defining multiobjective evolutionary algorithms and preference incorporation,” IEEE Trans. Evol. Comput., vol. 14, no. 4, pp. 636–664, Aug. 2010.

[30] M. Gong, F. Liu, W. Zhang, L. Jiao, and Q. Zhang, “Interactive MOEA/D for multi-objective decision making,” in Proc. 15th Annu. Genet. Evol. Comput. Conf. (GECCO), 2011, pp. 721–728.

[31] C. M. Fonseca and P. J. Fleming, “Multiobjective optimization and multiple constraint handling with evolutionary algorithms, i. a. unified formulation,” IEEE Trans. Syst., Man, Cybern. A, Syst., Humans, vol. 28, no. 1, pp. 26–37, Jan. 1998.

[32] K. Deb, J. Sundar, U. Bhaskara, and S. Chaudhuri, “Reference point based multiobjective optimization using evolutionary algorithms,” Int. J. Comput. Intell. Res., vol. 2, no. 3, pp. 273–286, 2006.

[33] L. Thiele, K. Miettinen, P. J. Korhonen, and J. M. Luque, “A preference-based evolutionary algorithm for multi-objective optimization,” Evol. Comput., vol. 17, no. 3, pp. 411–436, 2009.

[34] L. B. Said, S. Bechikh, and K. Ghédira, “The r-dominance: A new dominance relation for interactive evolutionary multicriteria decision making,” IEEE Trans. Evol. Comput., vol. 14, no. 5, pp. 801–818, Oct. 2010.

[35] K. Li, R. Chen, G. Min, and X. Yao, “Integration of preferences in decomposition multiobjective optimization,” IEEE Trans. Cybern., vol. 48, no. 12, pp. 3359–3370, Dec. 2018.

[36] J. Branke, T. Kaufler, and H. Schmeck, “Guidance in evolutionary multi-objective optimization,” Adv. Eng. Softw., vol. 32, no. 6, pp. 499–507, 2001.

[37] K. Deb, “Multi-objective evolutionary algorithms: Introducing bias among Pareto-optimal solutions,” in Advances in Evolutionary Computing. A. Ghosh and S. Tsutsui, Eds. Berlin, Germany: Springer, 2003, pp. 263–292.

[38] E. Zitzler, D. Brockhoff, and L. Thiele, “The Hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration,” in Proc. 4th Int. Conf. Evol. Multi-Criterion Optim. (EMO), 2006, pp. 862–876.
T. Wagner and H. Trautmann, “Integration of preferences in Hypervolume-based multiobjective evolutionary algorithms by means of desirability functions,” IEEE Trans. Evol. Comput., vol. 14, no. 5, pp. 688–701, Oct. 2010.

M. Köksalan and I. Karahan, “An interactive territory defining evolutionary algorithm: ITDEA,” IEEE Trans. Evol. Comput., vol. 14, no. 5, pp. 702–722, Oct. 2010.

H. Takagi, “Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation,” Proc. IEEE, vol. 89, no. 9, pp. 1275–1296, Sep. 2001.

R. Deveaud, J. Mothe, M. Z. Ullah, and J. Nie, “Learning to adaptively rank document retrieval system configurations,” ACM Trans. Inf. Syst., vol. 7, no. 1, pp. 1–41, 2019.

J. Pessiot, T.-Y. Truong, N. Usunier, M.-R. Amini, and P. Gallinari, “Learning to rank for collaborative filtering,” in Proc. 9th Int. Conf. Enterprise Inf. Syst. (ICEIS), 2007, pp. 145–151.

T.-Y. Liu, Learning to Rank for Information Retrieval. Berlin, Germany: Springer, 2011.

M. Karimzadehgan, W. Li, R. Zhang, and J. Mao, “A stochastic learning-to-rank algorithm and its application to contextual advertising,” in Proc. 20th Int. Conf. World Wide Web (WWW), 2011, pp. 377–386.

C. J. C. Burges et al., “Learning to rank using gradient descent,” in Proc. 22nd Int. Conf. Mach. Learn. (ICML), vol. 119, 2005, pp. 89–96.

L. Bottou, “Stochastic gradient descent tricks,” in Neural Networks: Tricks of the Trade (Lecture Notes in Computer Science 7700), 2nd ed., G. Montavon, G. B. Orr, and K. Müller, Eds. Heidelberg, Springer: Berlin, Germany, 2012, pp. 421–436.

K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, “Scalable test problems for evolutionary multiobjective optimization,” in Evolutionary Multiobjective Optimization (Advanced Information and Knowledge Processing). London, U.K.: Springer, 2005, pp. 105–145.

I. Das and J. E. Dennis, “Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems,” SIAM J. Optim., vol. 8, no. 3, pp. 631–657, 1998.

K. Cioimek and M. Kądziński, “Polyrun: A java library for sampling from the bounded convex polytopes,” SoftwareX, vol. 13, Jan. 2021, Art. no. 100659.

T. Tervonen, G. van Valkenhoef, N. Bastürk, and D. Postmus, “Hit-and-run enables efficient weight generation for simulation-based multiple criteria decision analysis,” Eur. J. Oper. Res., vol. 224, no. 3, pp. 552–559, 2013.

M. P. Hansen and A. Jaszkiewicz, “Evaluating the quality of approximations of the non-dominated set,” Inst. Math. Model., Tech. Univ. Denmark, Lyngby, Denmark, Rep. IMM-REP-1998-7, Jul. 1987.

K. Shang, H. Ishibuchi, and X. Ni, “R2-based hypervolume contribution approximation,” IEEE Trans. Evol. Comput., vol. 24, no. 1, pp. 185–192, Feb. 2020.

T. Wagner, H. Trautmann, and D. Brockhoff, “Preference articulation by means of the R2 indicator,” in Proc. 7th Int. Conf. Evol. Multi-Criterion Optim. (EMO), vol. 7811, 2013, pp. 81–95.

H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima, “Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes,” IEEE Trans. Evol. Comput., vol. 21, no. 2, pp. 169–190, Apr. 2017.

Z. Wang, Y.-S. Ong, and H. Ishibuchi, “On scalable multiobjective test problems with hardly dominated boundaries,” IEEE Trans. Evol. Comput., vol. 25, no. 2, pp. 217–231, Apr. 2019.

H. Ishibuchi, H. Masuda, and Y. Nojima, “Pareto fronts of many-objective degenerate test problems,” IEEE Trans. Evol. Comput., vol. 20, no. 5, pp. 807–813, Oct. 2016.

K. Deb and R. B. Agrawal, “Simulated binary crossover for continuous search space,” Complex Syst., vol. 9, no. 3, pp. 115–148, 1994.

K. Deb and S. Agrawal, “A niched-penalty approach for constraint handling in genetic algorithms,” in Proc. Int. Conf. Artif. Neural Nets Genet. Algorithms (ICANNGA), 1999, pp. 235–243.

K. Li, K. Deb, and X. Yao, “R-metric: Evaluating the performance of preference-based evolutionary multiobjective optimization using reference points,” IEEE Trans. Evol. Comput., vol. 22, no. 6, pp. 821–835, Dec. 2018.

F. Wilcoxon, “Individual comparisons by ranking methods,” Biometrics Bull., vol. 1, no. 6, pp. 80–83, 1945.

J. Derrac, S. García, D. Molina, and F. Herrera, “A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms,” Swarm Evol. Comput., vol. 1, no. 1, pp. 3–18, 2011.

N. Mittas and L. Angelis, “Ranking and clustering software cost estimation models through a multiple comparisons algorithm,” IEEE Trans. Softw. Eng., vol. 39, no. 4, pp. 537–551, Apr. 2013.

A. Vargha and H. D. Delaney, “A critique and improvement of the CL common language effect size statistics of McGraw and Wong,” J. Educ. Behav. Stat., vol. 25, no. 2, pp. 101–132, 2000.

K. Jørvelin and J. Kekäläinen, “IR evaluation methods for retrieving highly relevant documents,” SIGIR Forum, vol. 51, no. 2, pp. 243–250, 2017.

P. Ren et al., “A survey of deep active learning,” ACM Comput. Surv., vol. 54, no. 9, pp. 1–180, 2022.

I. Covert, S. M. Lundberg, and S. Lee, “Explaining by removing: A unified framework for model explanation,” J. Mach. Learn. Res., vol. 22, pp. 1–209, Jan. 2021.

H. Müller and A. Sedley, “Designing surveys for HCI research,” in Proc. Extended Abstracts Human Factors Comput. Syst. (CHI), New York, NY, USA, 2015, pp. 2485–2486.

Ke Li (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science and technology from Xiangtan University, Xiangtan, China, in 2007 and 2010, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2015.

His current research interests include the evolutionary multiobjective optimization, multi-criteria decision making, machine learning, and real-world applications, especially to software engineering.

Dr. Li is a recipient of the prestigious UKRI Future Leaders Fellowship, Turing Fellowship, and an Amazon Research Award. He is currently serving as an Associate Editor for IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and three other academic journals.

Guiyu Lai received the B.Sc. degree in computer science from Nanjing University of Science and Technology, Nanjing, China, in 2019. He is currently pursuing the M.Sc. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China.

His current research interest is evolutionary multiobjective optimization.

Xin Yao (Fellow, IEEE) received the B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1982, the M.Sc. degree from the North China Institute of Computing Technologies, Beijing, China, in 1985, and the Ph.D. degree from USTC in 1990.

He is a Chair Professor of Computer Science with the Southern University of Science and Technology, Shenzhen, China, and a part-time Professor of Computer Science with the University of Birmingham, Birmingham, U.K. His current research interests include evolutionary computation, machine learning, and their real-world applications, especially to software engineering.

Dr. Yao was a recipient of the prestigious Royal Society Wolfson Research Merit Award in 2012, the IEEE Computational Intelligence Society (CIS) Evolutionary Computation Pioneer Award in 2013, and the IEEE Frank Rosenblatt Award in 2020. His work won the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010, 2016, and 2017 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Awards, the 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and many other best paper awards. He was the President of IEEE CIS from 2014 to 2015 and the Editor-in-Chief of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008. He was a Distinguished Lecturer of IEEE CIS.